UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE QUIXADÁ

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DAVID ALVES SOARES

TURING STATION: UMA PLATAFORMA WEB PARA ESTUDO E SIMULAÇÃO DE MÁQUINAS DE TURING

QUIXADÁ

2025

DAVID ALVES SOARES

TURING STATION: UMA PLATAFORMA WEB PARA ESTUDO E SIMULAÇÃO DE MÁQUINAS DE TURING

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Dr. Paulo de Tarso Guerra Oliveira.

QUIXADÁ

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S653t Soares, David Alves.
Turing Station: uma plataforma web para estudo e simulação de máquinas de Turing / David Alves Soares. – 2025.
88 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2025.
Orientação: Prof. Dr. Paulo de Tarso Guerra Oliveira.

1. Máquina de Turing. 2. Aplicação web. 3. Aprendizado prático. 4. Ferramentas educacionais. I. Título.
CDD 004

DAVID ALVES SOARES

TURING STATION: UMA PLATAFORMA WEB PARA ESTUDO E SIMULAÇÃO DE MÁQUINAS DE TURING

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: 28/02/2025.

BANCA EXAMINADORA

Prof. Dr. Paulo de Tarso Guerra Oliveira (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Me. Carlos Roberto Rodrigues Filho
Universidade Federal do Ceará (UFC)

Prof. Dr. Alexandre Matos Arruda
Universidade Federal do Ceará (UFC)

Dedico este trabalho a Deus, por ter me concedido forças para superar os desafios. À minha família e à minha namorada, pelo apoio constante e pela presença em cada etapa desta grande conquista.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, que me concedeu força, fé e perseverança para superar todos os obstáculos que surgiram ao longo desta jornada.

Aos meus pais, Gláucio e Cristiana, e aos meus irmãos, Bruno, Breno e Lucas, que sempre acreditaram no meu potencial e me incentivaram a concluir esta graduação e a realizar os meus sonhos. Jamais esquecerei o apoio e a confiança que depositaram em mim. À minha namorada, Nayana, que esteve ao meu lado nos melhores e piores momentos, e que, com paciência e carinho, sempre me escutou e me motivou a seguir em frente, mesmo diante das dificuldades.

Aos meus amigos da universidade, que tornaram os momentos difíceis mais leves e contribuíram para que eu levasse essa caminhada acadêmica com serenidade. Aos meus amigos da minha terra natal, que, mesmo à distância, celebraram comigo cada conquista alcançada.

Ao meu orientador, Prof. Dr. Paulo de Tarso Guerra Oliveira, agradeço pela paciência em discutir minhas ideias, pela compreensão das minhas limitações e pelo apoio constante, que tornou o processo de escrita deste trabalho mais tranquilo. Com sua orientação, alcancei metas que, antes, pareciam distantes. Sou também grato aos membros da banca examinadora, que enriqueceram este trabalho com valiosas sugestões.

Agradeço ainda a todos os integrantes da Universidade Federal do Ceará – Campus Quixadá, que tornam o ambiente universitário tão acolhedor e inspirador. Por fim, deixo minha gratidão a todos que, direta ou indiretamente, contribuíram para a minha formação.

"Se eu vi mais longe, foi por estar sobre ombros de gigantes."
(Isaac Newton)

RESUMO

A Máquina de Turing é um conceito de extrema importância para a Teoria da Computação. Trata-se de um modelo fundamental no campo da computabilidade, área que investiga o que pode ou não ser computado por meio de algoritmos. Para facilitar o entendimento desse conceito, diversas ferramentas tecnológicas foram desenvolvidas com o objetivo de ajudar estudantes a aprender enquanto praticam. No entanto, grande parte dessas ferramentas apresenta limitações, seja pela falta de funcionalidades, problemas de usabilidade ou interfaces pouco organizadas e pouco atraentes para os usuários. Este trabalho desenvolve, apresenta e avalia a Turing Station, uma aplicação web responsiva criada para facilitar o aprendizado prático dos conceitos de Máquinas de Turing. A Turing Station permite a criação e simulação interativa de Máquinas de Turing e suas variantes, sendo acessível em diversos dispositivos diferentes. A validação da Turing Station foi realizada por meio de um formulário aplicado a um grupo de estudantes após o uso real da aplicação para resolução de exercícios relacionados a Máquinas de Turing, na disciplina de Teoria da Computação, na Universidade Federal do Ceará – Campus Quixadá. Os resultados demonstraram alta satisfação por parte dos usuários, que destacaram a facilidade de uso, a interface bem desenvolvida e, principalmente, o impacto positivo da ferramenta no entendimento e na prática dos conceitos de Máquinas de Turing abordados em sala de aula.

Palavras-chave: máquina de Turing; aplicação web; aprendizado prático; ferramentas educacionais.

ABSTRACT

The Turing Machine is an extremely important concept for the Theory of Computation. This is a fundamental model in the field of computability, an area that investigates what can and cannot be computed through algorithms. To facilitate the understanding of this concept, several technological tools have been developed with the aim of helping students learn while practicing. However, most of these tools have limitations, whether due to a lack of functionality, usability problems or poorly organized and unattractive interfaces for users. This work develops, presents and evaluates Turing Station, a responsive web application created to facilitate practical learning of Turing Machine concepts. Turing Station allows the creation and interactive simulation of Turing Machines and their variants, being accessible on several different devices. Validation of the Turing Station was carried out using a form applied to a group of students after actually using the application to solve exercises related to Turing Machines, in the Theory of Computation classes, at the Federal University of Ceará – Campus Quixadá. The results demonstrated high satisfaction on the part of users, who highlighted the ease of use, the well-developed interface and, mainly, the positive impact of the tool on the understanding and practice of Turing Machine concepts covered in the classroom.

Keywords: Turing machine; web application; practical learning; educational tools.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

E-Learning	*Eletronic Learning*
CSS	*Cascading Style Sheets*
FLApp	*Formal Languages and Automata Application*
HTML	*Hypertext Markup Language*
JFLAP	*Java Formal Languages and Automata Package*
MT	Máquina de Turing
MTND	Máquina de Turing Não Determinística
RF	Requisitos Funcionais
RNF	Requisitos Não Funcionais
SUS	Escala de Usabilidade do Sistema
UI	Interface de Usuário
UX	Experiência de Usuário

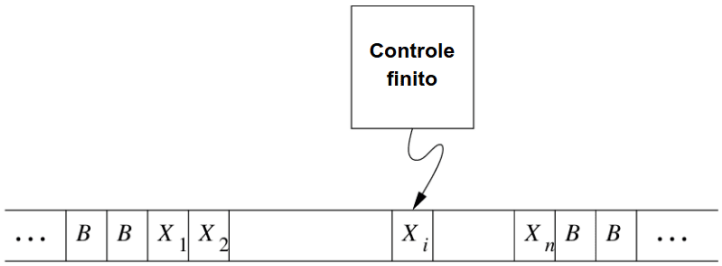
SUMÁRIO

1 INTRODUÇÃO

Máquina de Turing (MT) é um dos modelos fundamentais no estudo de Teoria da Computação. A Teoria da Computação é o ramo da Ciência da Computação que busca compreender quais problemas podem ser computados por uma determinada máquina abstrata ou algoritmo (Hopcroft *et al.*, 2006; Sipser, 2012). A MT foi inicialmente descrita por Alan Turing (Turing, 1936) como um dispositivo computacional abstrato destinado a investigar as limitações da computabilidade, ou seja, entender o que é ou não computável (Linz; Rodger, 2022).

A MT é informalmente descrita como um tipo de máquina de estados que possui uma fita discreta infinita e uma cabeça de fita que pode ler, escrever, e mover-se sobre esta fita. A fita inicialmente contém apenas os símbolos a serem computados. A máquina fará computações, escrevendo ou apagando símbolos na fita utilizando sua cabeça de fita e produzirá uma saída apenas se entrar em algum estado de aceitação ou a sequência de símbolos de entrada for rejeitada, caso contrário, executará para sempre (Hopcroft *et al.*, 2006). A Figura 1 apresenta um exemplo de MT.

Figura 1 – Exemplo de uma Máquina de Turing.



Fonte: Adaptado de Hopcroft *et al.* (2006, p. 326).

No contexto da Universidade Federal do Ceará - Campus Quixadá, máquinas de Turing são ensinadas por professores utilizando, principalmente, exposição oral e quadro branco, apoiado por referências a livros didáticos e atividades-exercício em papel. Isso por vezes pode ser abstrato e difícil de compreender para muitos estudantes. Esse método tradicional, embora efetivo em muitos casos, frequentemente não proporciona uma compreensão prática e profunda do conceito apresentado.

Para Haleem *et al.* (2022), as metodologias tradicionais de ensino falham em criar um ambiente de aprendizagem que gere mais envolvimento, dinamicidade e que tenha avaliações mais imediatas. Em contraste a isso está o uso de ferramentas digitais de ensino, que podem promover muitos benefícios para os professores e alunos, principalmente quando utilizadas em

conjunto com outras metodologias já existentes.

Um exemplo dos efeitos relacionados ao uso de ferramentas digitais no ensino foi demonstrado em um estudo que testou a plataforma digital para aprendizado de idiomas *Duolingo*[1] com estudantes do idioma espanhol com idades de até 18 anos. Este estudo demonstrou que os estudantes obtiveram uma melhoria significativa nas suas habilidades linguísticas após o uso da ferramenta. O trabalho também confirmou que o principal fator para tal melhoria foi a motivação dos estudantes ao utilizarem a plataforma (Vesselinov; Grego, 2012).

A integração de tecnologias digitais na educação pode tornar o ambiente de ensino mais agradável, dinâmico, engajador e inspirador para os estudantes. Isso permite que eles mantenham-se interessados em aprender determinados assuntos com menos distrações. Somado a isso, as tecnologias digitais no ensino podem ajudar os estudantes a desenvolverem habilidades requisitadas na vida profissional, como compreensão de problemas e pensamento criativo (Haleem *et al.*, 2022).

Neste trabalho analisamos algumas ferramentas digitais para auxílio ao estudo de MTs, sendo essas as ferramentas apresentadas nos trabalhos de Procopiuc *et al.* (1996), Pereira e Terra (2018) e Vasconcelos e Guerra (2023). Essa análise mostrou que, embora atendam ao objetivo de auxiliar no estudo de MT, essas ferramentas apresentam diversos problemas que tornam a utilização das mesmas menos atrativa para os usuários. Esses problemas estão ligados principalmente a Interface de Usuário (UI, do inglês *User Interface*), que representa os elementos visuais da interface, e a Experiência de Usuário (UX, do inglês *User Experience*), que representa a experiência geral do usuário ao utilizar a aplicação.

O trabalho de Pereira e Terra (2018) apresenta um aplicativo móvel *Android* para estudo de diversos conteúdos de Teoria da Computação, incluindo MTs, denominado *Formal Languages and Automata Application* (FLApp)[2]. O trabalho de Vasconcelos e Guerra (2023) apresenta uma biblioteca denominada Teocomp[3], que pode ser utilizada para o aprendizado e ensino de MTs por meio da ferramenta *Jupyter Notebooks*. O trabalho de Procopiuc *et al.* (1996) apresenta a ferramenta amplamente conhecida e utilizada *Java Formal Languages and Automata Package* (JFLAP)[4], esta ferramenta também possui diversos recursos que podem ajudar no estudo e no ensino de estruturas de Teoria da Computação, inclusive MTs, podendo ser utilizada por meio de dispositivos *desktops*.

[1] https://pt.duolingo.com/
[2] https://github.com/rterrabh/LFApp
[3] https://github.com/daviromero/teocomp/
[4] https://www.jflap.org/jflaptmp/

Mesmo dispondo de diversas funcionalidades, tais ferramentas possuem alguns problemas relacionados a UI e/ou UX. Algumas delas precisam ser utilizadas via código, ou seja, o usuário precisa estar familiarizado com a linguagem de programação ou a linguagem descritiva adotada pelos criadores, como por exemplo a biblioteca Teocomp; outras permitem apenas a utilização da aplicação em um único tipo de dispositivo ou sistema operacional, como por exemplo o aplicativo FLApp; além do próprio JFLAP, que possui diversos problemas relacionados a UI e UX, como mostrado em uma avaliação de experiência de uso feita em Silva *et al.* (2023).

Este trabalho teve por finalidade desenvolver e avaliar uma aplicação web, denominada Turing Station. Esta aplicação possui funcionalidades focadas no apoio do estudo de MTs. Para a realização deste trabalho, foram avaliadas diversas ferramentas educacionais livres para o estudo de MTs, visando mesclar as melhores funcionalidades e pontos de usabilidade das mesmas com a aplicação proposta.

A aplicação foi testada e avaliada por alunos matriculados na disciplina de Teoria da Computação na Universidade Federal do Ceará - Campus Quixadá, no segundo semestre de 2024, após a utilização da mesma para a resolução de alguns problemas sobre MTs em uma lista de exercícios. Após isso, por meio de um formulário, foram coletadas as opiniões dos estudantes sobre os benefícios que a ferramenta trouxe para o seu aprendizado, além de reunir opiniões sobre melhorias e adição de novas funcionalidades para a plataforma.

Através da Turing Station, os alunos puderam criar, visualizar e executar MTs e suas variantes de forma dinâmica e muito intuitiva. Além disso, os estudantes puderam importar e exportar arquivos de máquinas anteriormente criadas, visualizar fila de execução, entre outros. A plataforma também conta com tutoriais e guias para orientá-los no processo de aprendizagem e no uso da ferramenta, além de conter um design atrativo para os alunos. Ademais, a plataforma foi disponibilizada de forma online e livre para qualquer usuário. Parte deste trabalho foi publicado em Alves e Guerra (2024) no 13° Congresso Brasileiro de Informática na Educação (CBIE 2024).

1.1 Contribuições

Este trabalho apresentou contribuições significativas para o contexto de estudo e prática dos conceitos de MTs. As principais contribuições são destacadas a seguir:

1. **Análise de ferramentas didáticas para estudo de MTs** - Este trabalho analisou de forma

empírica três ferramentas computacionais com objetivos similares para entender suas principais características, funcionalidades, e possíveis áreas de melhorias, o que pode ser utilizado como referência para outros estudos futuros na área.

2. **Construção de uma aplicação web para estudos e prática de MTs** - Este trabalho construiu uma aplicação web com o intuito de facilitar a prática, o ensino e o aprendizado dos conceitos de MTs. Tal aplicação pode ser melhorada e utilizada como objeto de estudo por trabalhos futuros na área.
3. **Avaliação dos impactos da aplicação desenvolvida** - Este trabalho também avaliou os impactos da aplicação na obteção de conhecimentos e melhoria do entendimento de alunos matriculados na disciplina de Teoria da Computação, por meio de um questionário respondido por um grupo de alunos após um período de uso.

1.2 Organização

Os próximos capítulos deste trabalho estão organizados da seguinte maneira: no Capítulo 2 são apresentados os principais conceitos que formaram a base para o desenvolvimento deste trabalho; no Capítulo 3 são apresentados alguns trabalhos relacionados a este trabalho; no Capítulo 4 são apresentadas algumas ferramentas semelhantes a aplicação proposta por este trabalho; no Capítulo 5 são detalhadas as etapas do desenvolvimento da aplicação proposta; no Capítulo 6 são detalhados os principais aspectos e as principais funcionalidades da aplicação proposta; no Capítulo 7 são apresentados os resultados obtidos ao avaliar a experiência de uso de um grupo de estudantes com a aplicação proposta; e por fim, o Capítulo 8 apresenta as conclusões obtidas e explora os possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os fundamentos teóricos necessários para a elaboração deste trabalho. A Seção 2.1 define a MT e as suas subseções descrevem as variantes de MTs que também farão parte da aplicação proposta por esse trabalho. Na Seção 2.2 são apresentados os conceitos teóricos por trás do uso de ferramentas digitais na educação.

2.1 Máquinas de Turing

Uma Máquina de Turing (MT) (Turing, 1936) é um dispositivo abstrato idealizado para investigar os limites da computabilidade, que é um campo da Teoria da Computação que busca compreender o que é ou não computável por máquinas abstratas e/ou algoritmos. A MT recebe como entrada uma palavra a ser computada e inicia sua computação. Neste contexto, entendemos uma palavra como uma sequência de símbolos finitos escolhidos dentre o conjunto de símbolos de um alfabeto qualquer. Por sua vez, um alfabeto é um conjunto finito e não vazio de símbolos (Hopcroft *et al.*, 2006; Sipser, 2012).

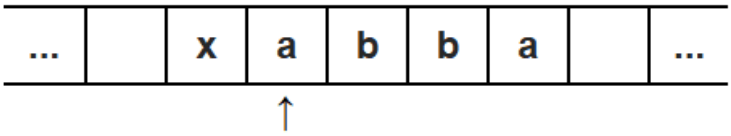
A MT é um tipo de máquina de estados e sua computação terá fim imediato sempre que ela se encontrar em um estado de aceitação ou não houver definida uma transição partindo do estado atual lendo o símbolo sob a cabeça de fita. Caso em nenhum momento da computação a MT chegue a um estado de aceitação ou rejeite a palavra, executará para sempre (Hopcroft *et al.*, 2006). Uma MT é uma máquina determinística, ou seja, para cada computação de um símbolo qualquer, sempre é possível saber o próximo estado onde a máquina irá se encontrar, tudo está determinado por suas funções de transição não ambíguas (Cohen, 1979; Hopcroft *et al.*, 2006; Sipser, 2012).

Ao entrar em um estado de aceitação, a MT reconhece a palavra computada como válida, neste caso, dizemos que ela aceita aquela palavra. Por outro lado, se para o símbolo lido na fita não houver definida uma transição partindo do estado atual da computação, a MT conclui que a palavra computada não é válida e a computação também terminará, no entanto, indicará que a palavra foi rejeitada pela MT (Hopcroft *et al.*, 2006).

A MT utiliza uma fita de tamanho infinito para armazenamento temporário de símbolos. Inicialmente, esta fita contém apenas a palavra a ser computada e infinitos espaços vazios (que também podem ser representados pelo símbolo □) à esquerda e à direita da palavra, e deve receber apenas um símbolo por vez em cada um de seus espaços. Associada a fita, está a

cabeça da fita, que pode mover-se para a esquerda e para a direita sobre a fita, e também pode ler e escrever um símbolo por vez. Qualquer entrada, processamento de palavras e saída de resultados será realizada sobre esta fita (Hopcroft *et al.*, 2006; Sipser, 2012). A Figura 2 mostra um exemplo de fita de uma MT com a palavra *xabba* e cabeça de fita sobre o primeiro *a*.

Figura 2 – Exemplo de fita de uma Máquina de Turing.



Fonte: Elaborada pelo autor.

A descrição formal de uma MT pode tomar vários formatos dependendo do autor escolhido. Para Hopcroft *et al.* (2006), uma MT pode ser descrita como uma séptupla no formato $(Q, \Sigma, \Gamma, \delta, q_0, \square, F)$, conforme a Definição 2.1.1.

Definição 2.1.1 *Uma Máquina de Turing M é uma 7-tupla* $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$*, onde:*

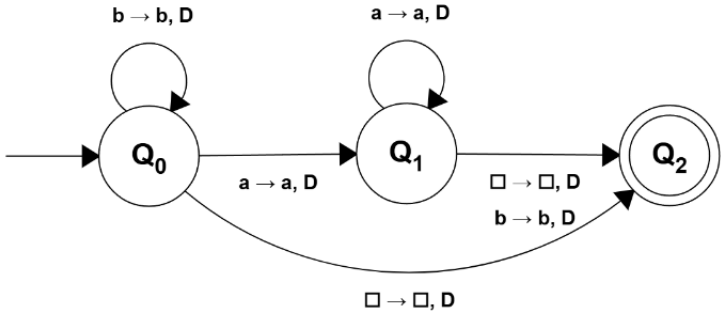
- Q *é um conjunto finito de estados;*
- Σ *é o alfabeto de entrada tal que* $\square \notin \Sigma$*;*
- Γ *é o alfabeto da fita, onde* $\Sigma \subseteq \Gamma$ *e* $\square \in \Gamma$*;*
- $\delta : Q' \times \Gamma \to Q \times \Gamma \times \{E, D\}$ *é a função de transição, onde* $Q' \subseteq Q - F$ *e* $\{E, D\}$ *representa as direções de movimento da cabeça de leitura/escrita, respectivamente, esquerda e direita;*
- $q_0 \in Q$ *é o estado inicial;*
- $\square$ *é o símbolo que representa uma célula vazia;*
- $F \subseteq Q$ *é o conjunto de estados de aceitação.*

Em conjunto com a representação da computação das MTs em formato de fita, muitas outras formas de representação também podem ser utilizadas. Um formato muito comum de representação é o diagramas de estados (Hopcroft *et al.*, 2006). Um diagrama de estados é uma forma visual de compreender os estados e as transições de um sistema. Este formato facilita a compreensão do comportamento que uma MT pode ter em um determinado estado da computação.

Na representação de uma MT por meio de um diagrama de estados, os estados são representados por círculos e as transições por setas rotuladas. Cada rótulo $A \to B, C$ em uma seta indica que a leitura do símbolo A desencadeia a escrita do símbolo B e um movimento na

direção C. O estado inicial é indicado por uma seta sem origem em outro estado. Os estados de aceitação são representados por círculos duplos. A Figura 3 ilustra a MT formalmente definida como $M = (\{Q_0, Q_1, Q_2\}, \{a, b\}, \{a, b, \square\}, \delta, Q_0, \square, \{Q_2\})$ onde $\delta(Q_0, a) = (Q_1, a, D)$, $\delta(Q_0, b) = (Q_0, b, D)$, $\delta(Q_0, \square) = (Q_2, \square, D)$, $\delta(Q_1, a) = (Q_1, a, D)$, $\delta(Q_1, b) = (Q_2, b, D)$, $\delta(Q_1, \square) = (Q_2, \square, D)$.

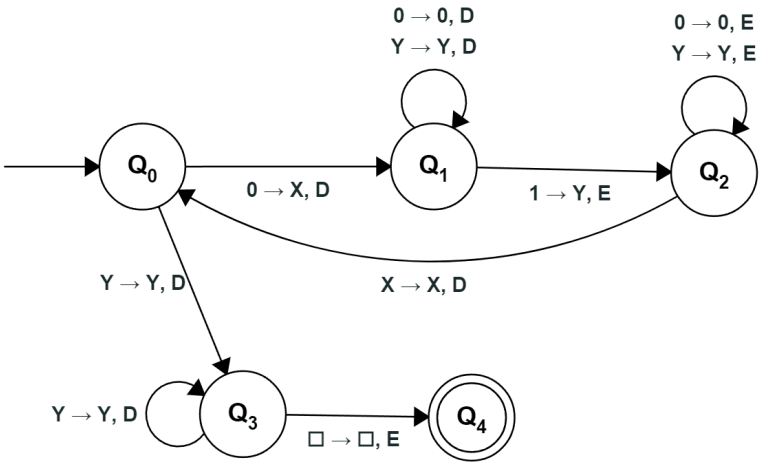
Figura 3 – Máquina de Turing representada por um diagrama de estados.



Fonte: Elaborada pelo autor.

Para compreender como a computação em uma MT acontece e como ela pode ser representada visualmente, suponha uma MT denominada M_1 (Figura 4) que testará se a palavra $S = 0011$ pertence a linguagem $L = \{0^n 1^n \mid n >= 1\}$, ou seja, a palavra é composta por um número de zeros seguidos pelo mesmo número de uns. Neste contexto, entendemos linguagem como um conjunto de palavras.

Figura 4 – MT M_1 que reconhece a linguagem L.



Fonte: Elaborada pelo autor.

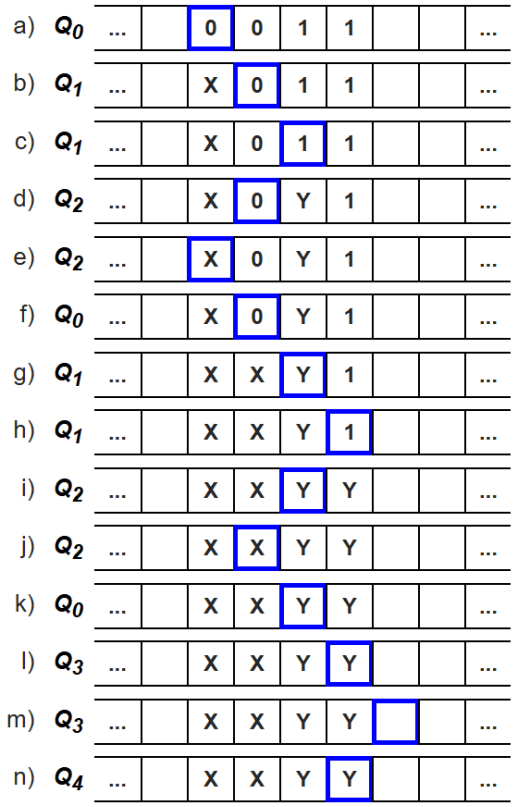
A M_1 iniciará sua computação no estado inicial Q_0 com a fita inicialmente na

configuração (a) da Figura 5. Observe que as configurações da fita ao decorrer da computação serão mostradas por meio dos itens presentes na Figura 5, cuja a posição da cabeça de fita a cada passo esta localizada na célula demarcada com bordas azuis. Ao decorrer de sua computação, M_1 lerá um símbolo de S por vez e seguirá os seguintes passos:

1. No estado Q_0, ao ler 0, escreverá X na posição demarcada da fita, moverá a cabeça de fita para a direita e mudará para o estado Q_1. A nova configuração da fita será (b).
2. No estado Q_1, ao ler 0, escreverá 0 na posição demarcada da fita, moverá a cabeça de fita para a direita e se manterá no estado Q_1. A nova configuração da fita será (c).
3. No estado Q_1, ao ler 1, escreverá Y na posição demarcada da fita, moverá a cabeça de fita para a esquerda e mudará para o estado Q_2. A nova configuração da fita será (d).
4. No estado Q_2, ao ler 0, escreverá 0 na posição demarcada da fita, moverá a cabeça de fita para a esquerda e se manterá no estado Q_2. A nova configuração da fita será (e).
5. No estado Q_2, ao ler X, escreverá X na posição demarcada da fita, moverá a cabeça de fita para a direita e mudará para o estado Q_0. A nova configuração da fita será (f).
6. Repita o passo 1. A nova configuração da fita será (g).
7. No estado Q_1, ao ler Y, escreverá Y na posição demarcada da fita, moverá a cabeça de fita para a direita e se manterá no estado Q_1. A nova configuração da fita será (h).
8. Repita o passo 3. A nova configuração da fita será (i).
9. No estado Q_2, ao ler Y, escreverá Y na posição demarcada da fita, moverá a cabeça de fita para a esquerda e se manterá no estado Q_2. A nova configuração da fita será (j).
10. Repita o passo 5. A nova configuração da fita será (k).
11. No estado Q_0, ao ler Y, escreverá Y na posição demarcada da fita, moverá a cabeça de fita para a direita e mudará para o estado Q_3. A nova configuração da fita será (l).
12. No estado Q_3, ao ler Y, escreverá Y na posição demarcada da fita, moverá a cabeça de fita para a direita e se manterá no estado Q_3. A nova configuração da fita será (m).
13. No estado Q_3, ao ler □, escreverá □ na posição demarcada da fita, moverá a cabeça de fita para a esquerda e mudará para o estado Q_4. Como o estado Q_4 é de aceitação, a computação termina, aceitando a palavra S. A configuração final da fita é (n).

No contexto da aplicação Turing Station, a representação das MTs se dará por uma combinação da representação da computação por fita e a representação da máquina por meio do diagrama de estados. Essa escolha se deve ao fator didático que essa combinação de representações proporciona. Além disso, na Universidade Federal do Ceará - Campus Quixadá,

Figura 5 – Passos da computação de M_1.



Fonte: Elaborada pelo autor.

os estudantes estão habituados a representação por meio de diagrama de estados devido a outras disciplinas que compõem o currículo acadêmico, como Engenharia de Software e Linguagens Formais e Autômatos, que exploram essa forma de representação em determinadas atividades.

2.1.1 Variantes de Máquinas de Turing

As variantes de Máquinas de Turing são definições alternativas para a noção de MT. Essas variantes adicionam novos conceitos e definições, como fita finita à esquerda, opção de permanecer com a cabeça de leitura parada após uma computação, não determinismo, e multifitas. Tais variantes podem, muitas vezes, ajudar na construção e no estudo de MTs mais complexas, devido a sua capacidade de abstrair e expandir certos conceitos. No entanto, mesmo essas variantes parecendo adicionar algum poder de computabilidade para MTs, elas não o fazem, pois é sempre possível construir uma MT clássica equivalente a qualquer uma de suas variantes (Hopcroft *et al.*, 2006; Sipser, 2012).

No contexto da disciplina de Teoria da Computação na Universidade Federal do Ceará - Campus Quixadá, essas e outras variantes são ensinadas em sala de aula para que os estudantes tenham noção da robustez que a MT possui. Essas variantes também podem ajudá-los

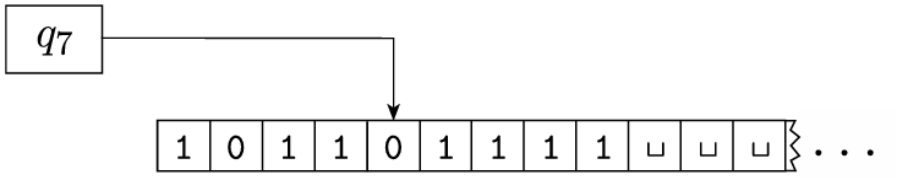
a elaborar máquinas maiores e mais complexas com um esforço menor do que seria necessário caso tentassem construí-las utilizando a MT clássica.

A aplicação proposta por este trabalho conta com ambientes para a prática tanto da MT clássica, como de cada uma das variantes de MTs citadas acima, pois essas máquinas são muito importantes para criar uma boa base de conhecimento em Teoria da Computação para os estudantes.

2.1.1.1 Máquina de Turing com fita finita à esquerda

A Máquina de Turing com fita finita à esquerda pode ser vista como uma variante dependendo do autor adotado. Por exemplo, a definição de Hopcroft *et al.* (2006), utilizada por este trabalho, fala que uma MT possui fita infinita para os dois lados, enquanto para Sipser (2012) a fita é finita à esquerda e infinita à direita. Embora a MT com fita finita à esquerda pareça mais limitada do que a MT com fita infinita para os dois lados, ela não é, pois ambas possuem o mesmo poder de computação. A Figura 6 apresenta um exemplo de uma MT com fita finita à esquerda segundo a definição de Sipser (2012).

Figura 6 – Fita de MT segundo a definição de Sipser (2012).



Fonte: Adaptado de Sipser (2012, p. 169).

2.1.1.2 Máquina de Turing com opção de parada

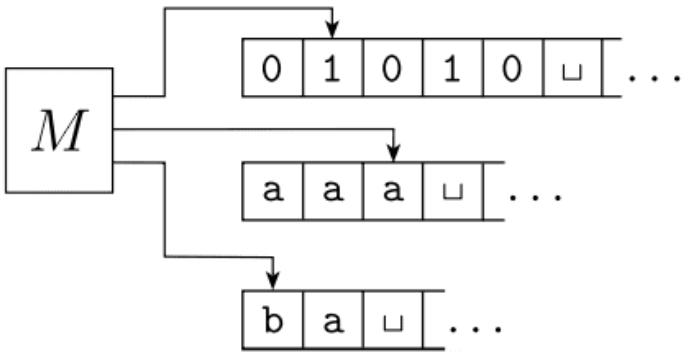
A Máquina de Turing com opção de parada é uma extensão da definição padrão de uma MT. Esta MT pode, ao realizar um passo da computação, movimentar sua cabeça de leitura para a esquerda, para a direita, ou permanecer parada. Formalmente, ela pode ser definida de maneira semelhante a MT clássica, tendo apenas uma variação no formato da sua função de transição, $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{\text{E, D, P}\}$, onde P representa o não movimento da cabeça de leitura. É válido destacar que a aceitação ou rejeição de uma palavra nesta variante de MT funciona exatamente como em uma MT clássica. Percebe-se que essa definição pode facilitar a elaboração de MTs, pois a opção de permanecer parada pode ser útil em determinadas situações

de computação, evitando a necessidade da criação de estados extras (Sipser, 2012; Linz; Rodger, 2022).

2.1.1.3 Máquina de Turing multifitas

A Máquina de Turing multifitas pode ser informalmente definida como sendo uma MT que possui múltiplas fitas, cada fita possuindo uma cabeça de leitura e escrita própria (veja um exemplo na Figura 7). No início, apenas a primeira fita conterá a palavra a ser computada e todas as outras fitas iniciarão vazias. A cada computação, a MT multifitas pode ler, escrever e mover-se para a esquerda ou para a direita de forma simultânea e isolada em cada fita, ou seja, cada fita poderá conter uma configuração diferente ao mesmo tempo (Hopcroft *et al.*, 2006).

Figura 7 – Múltiplas fitas de uma MT multifitas.



Fonte: Sipser (2012, p. 177).

A computação de uma MT multifitas será bem sucedida quando ao computar uma palavra, seguindo as regras de transição, a MT chegar a um estado de aceitação. Se para o símbolo lido não houver uma transição definida no conjunto de regras de transição, ela rejeitará a palavra. Se a MT em nenhum momento entrar em um estado de aceitação ou rejeitar a palavra, executará para sempre (Hopcroft *et al.*, 2006).

Formalmente, uma MT multifitas pode ser definida de forma semelhante a uma MT clássica, modificando apenas a estrutura da função de transição, que ficará $\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{\text{E, D}\}^k$, onde k representa o seu número de fitas (Hopcroft *et al.*, 2006).

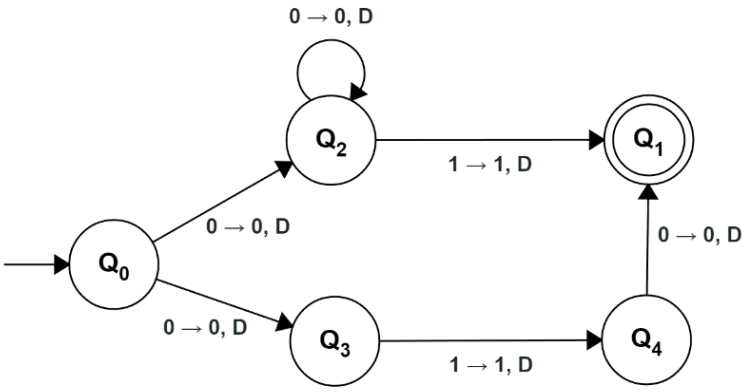
2.1.1.4 Máquina de Turing não determinística (MTND)

Uma Máquina de Turing Não Determinística (MTND) é uma MT que faz o uso do não determinismo. O não determinismo, na teoria da computabilidade, é definido como

sendo a possibilidade de uma máquina abstrata poder estar em vários estados ao mesmo tempo ao processar um determinado símbolo. Uma máquina não determinística tem a possibilidade de criar vários ramos paralelos de computação, ou seja, várias "cópias" da máquina são feitas sempre que houver mais de uma possibilidade de estado de destino com um determinado símbolo lido (Cohen, 1979; Hopcroft *et al.*, 2006; Sipser, 2012).

A Figura 8 apresenta um exemplo de MTND. A MTND M_{nd} testará se uma palavra qualquer pertence a linguagem $L_{nd} = \{010\} \bigcup \{00^*1\}$. O símbolo *, neste contexto, é chamado de operador de Kleene e é usado para denotar zero ou mais ocorrências do elemento precedente, ou seja, 00^*1 indica uma ou mais ocorrências de 0 seguido de um único 1. Observe que, ao iniciar a computação e ler "0", a M_{nd} computará em paralelo dois caminhos possíveis.

Figura 8 – MTND M_{nd} que reconhece a linguagem L_{nd}.

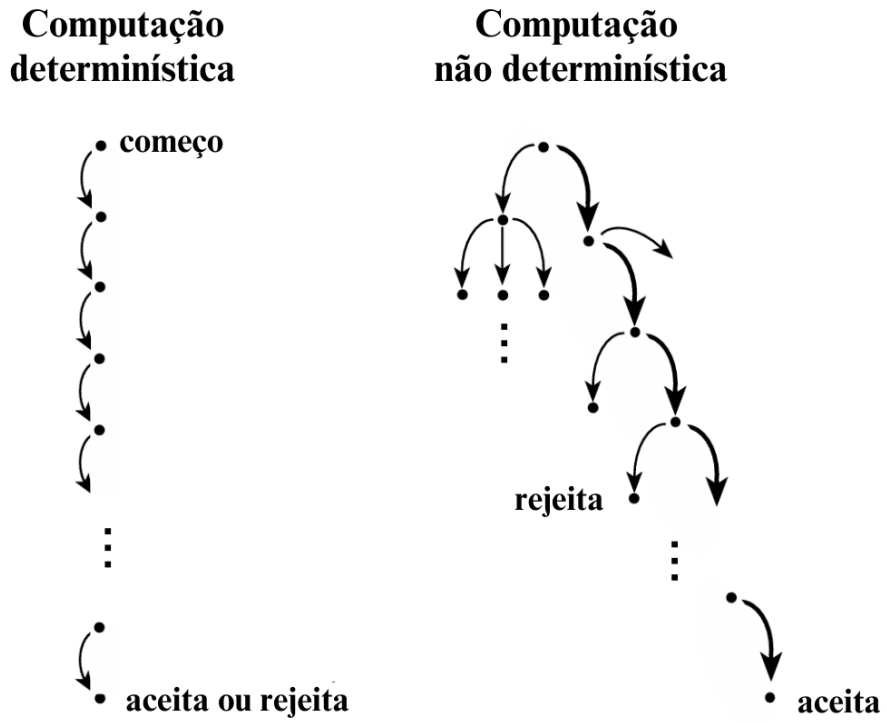


Fonte: Elaborada pelo autor.

A computação de uma MTND terminará quando pelo menos uma de suas ramificações chegar a um estado de aceitação (aceitando a palavra computada) ou todas as suas ramificações tiverem transições que não estão definidas no conjunto de regras de transição (rejeitando a palavra computada). Caso nenhuma das duas possibilidades anteriores aconteça, a MTND executará para sempre (Hopcroft *et al.*, 2006). A Figura 9 representa, de forma abstrata, a comparação entre o fluxo de computação de uma MT e as ramificações do fluxo de computação de uma MTND.

Formalmente, a MTND também tem sua definição parecida com uma MT clássica, modificando apenas o formato da função de transição, que se torna $\delta : Q \times \Gamma \rightarrow P(Q \times \Gamma \times \{\text{E, D}\})$, onde $P(A) = \{X \mid X \subseteq A\}$ representa o conjunto das partes (Hopcroft *et al.*, 2006).

Figura 9 – Comparação de fluxo de computação entre uma MT e uma MTND.



Fonte: Adaptado de Sipser (2012, p. 49).

2.2 Ferramentas digitais na educação

Ferramentas digitais na educação fazem parte de um modelo de educação conhecido como *Eletronic Learning* (*E-Learning*). Neste modelo, os conteúdos ensinados, atividades, e *feedback* são disponibilizados tipicamente de forma *on-line*, ou seja, requer *internet* para acesso pelos estudantes e pelas instituições que os monitoram (Devedžic, 2006). Muitos são os meios digitais em que o *E-Learning* pode-se estabelecer, sendo alguns dos principais: jogos *on-line*, salas de aula virtuais, tutoriais, plataformas de simulações, entre outros (Clark; Mayer, 2023).

O ensino por meio de ferramentas digitais pode proporcionar a separação de professores e alunos. Segundo Clark e Mayer (2023), este modelo de ensino pode se estabelecer de forma síncrona, quando os participantes interagem em tempo real por algum meio tecnológico, como videoconferências ou aulas remotas ao vivo; ou de forma assíncrona, quando o aprendizado não acontece em tempo real, ou seja, o estudante acessa materiais online, como videoaulas gravadas, slides, ou até mesmo plataformas de simulação, em seu próprio tempo.

Para Gautam e Tiwari (2016), esse modelo de educação pode trazer diversos benefícios para os estudantes, alguns deles são: flexibilidade de tempo e espaço para que os estudantes estudem onde e quando quiserem; os estudantes podem levar o próprio ritmo para aprender pois

o conteúdo estará disponível por bastante tempo; interatividade entre instituição e alunos, o que pode melhorar a resolução de problemas que os estudantes possam vir a ter; proporciona confiança no aprendizado, pois o conteúdo estará disponível para que os estudantes revejam sempre que necessitarem.

Atualmente, um dos principais veículos proporcionadores de educação por meio de ferramentas digitais é a web (Devedžic, 2006). A web é a rede mundial que interliga computadores em todo o mundo através da *internet*. Dentro da web, é possível encontrar páginas de conteúdo com textos, imagens, vídeos e muito mais, chamadas de páginas web, ou até mesmo agrupamentos dessas páginas, os quais são denominados *websites* (Choudhury, 2014).

Segundo Allison *et al.* (2012), a utilização de ferramentas educacionais web pode ser considera uma revolução na educação, pois a web proporciona um um vasto mundo de possibilidades para o contexto educacional. Ela oferece facilidades no design, criação, adaptação e testes de ferramentas que podem auxiliar no ensino. Além disso, a web ajuda na análise das estratégias e da qualidade do serviço. Isso contribui para uma melhor personalização e construção da infraestrutura educacional proposta.

Para Elmunsyah *et al.* (2020), é necessário considerar aspectos da Experiência de Usuário (UX) e Interface de Usuário (UI) para que as ferramentas de ensino *on-line* atinjam os objetivos esperados. Ele argumenta que o cumprimento de um objetivo pessoal, ao utilizar uma ferramenta, é o centro de toda a experiência e que as qualidades da UI do produto/serviço também contribuem de forma positiva para essa boa experiência, principalmente ao facilitar a sua usufruição.

Guntupalli (2008) cita alguns aspectos que contribuem para a determinação da qualidade da experiência de uso de uma ferramenta digital, sendo eles:

- Confiabilidade: a aplicação deve se comportar de forma padrão e precisa, do começo ao fim;
- Eficiência: a aplicação precisa executar da melhor forma utilizando a menor quantidade de recursos computacionais possível;
- Concisão: a aplicação deve utilizar apenas informações necessárias para executar, não necessitando de informações extras;
- Portabilidade: a aplicação deve poder ser executada em diversos dispositivos diferentes (ex: computadores de mesa, *smartphones, tablets*);
- Consistência: a aplicação deve manter todos os padrões de código, texto, formatos, cores,

ao decorrer de toda a sua utilização, para não causar estranhamento ao usuário;

- Capacidade de manutenção: a aplicação precisa dispor de uma facilidade de atualização, seja adicionar novos recursos, modificar recursos existentes, ou apenas atualizar seu código para versões mais atuais;
- Compreensibilidade: a aplicação precisa ser compreensível para o usuário, não apenas utilizável.

Ainda segundo Guntupalli (2008), juntamente aos aspectos de qualidade geral da aplicação, deve-se levar em consideração a sua UI, visto que ela funciona como a face frontal da aplicação, sendo a primeira coisa que o usuário vê ao utilizá-la. A UI precisa apresentar informações suficientes para que o usuário compreenda tudo o que a aplicação tem a oferecer. Para ele, os *designers* devem levar em consideração três aspectos diferentes ao construir a UI de uma aplicação: interface, informação e interação.

A interface deve ser construída levando em consideração a harmonia de cores pois isso ajudará na identificação dos elementos mais facilmente, além de ser necessário considerar as pessoas com problemas para enxergar certas cores. Botões, menus, ícones e navegação padronizada também devem ser utilizados. As informações devem estar dispostas de forma apropriada, bem divididas, considerando a apresentação de textos, ícones e imagens, para que o usuário entenda bem o que aquela interface pretende mostrá-lo. Já quando se trata de interação, deve-se levar em consideração o público alvo da aplicação. Deve-se facilitar a localização de informações sobre como utilizar cada aspecto da plataforma pelo usuário, além de, dependendo do tipo de aplicação, utilizar-se de atalhos de teclado para otimizar a utilização da aplicação (Guntupalli, 2008).

3 TRABALHOS RELACIONADOS

Este capítulo tem por objetivo apresentar e comparar alguns trabalhos que possuem semelhanças com o trabalho proposto, citando algumas de suas contribuições para este trabalho. Por fim, a Seção 3.4 faz uma análise comparativa entre os trabalhos relacionados e o trabalho porposto.

3.1 *Increasing engagement in automata theory with JFLAP*

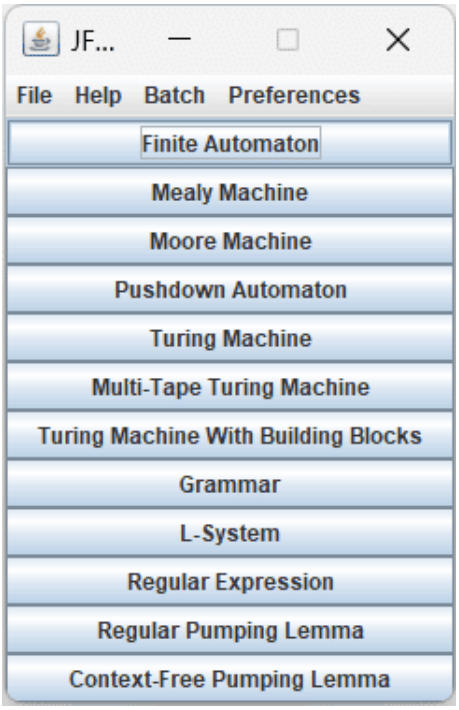
No trabalho Rodger *et al.* (2009), foi avaliada a aplicação *Java Formal Languages and Automata Package* (JFLAP) em diversas universidades, em turmas que abordam conteúdos de Teoria da Computação, por um período de 2 anos. O JFLAP é a reconstrução de um *software* um pouco mais antigo, denominado *Formal Languages and Automata Package* (FLAP), descrito no trabalho LoSacco e Rodger (1993), utilizando a linguagem de programação *Java*.

O JFLAP é um *software desktop* que permite a simulação de diversas estruturas abstratas de Teoria da Computação. O trabalho Procopiuc *et al.* (1996) apresenta de maneira mais abrangente uma das primeiras versões do JFLAP. Em seu lançamento, por volta de 1996, o JFLAP possuía apenas algumas funcionalidades para estudo de conteúdos ligados a Teoria da Computação. Atualmente o JFLAP é uma das ferramentas mais completas e utilizadas para simulação digital de autômatos e máquinas abstratas, possuindo diversas funcionalidades que vão desde a conversão entre tipo de autômatos, criação de máquinas de Turing, criação de máquinas de Mealy, criação de máquinas de Moore, manipulação de gramáticas, e muito mais, como é possível visualizar na Figura 10.

No trabalho Rodger *et al.* (2009), a avaliação do JFLAP foi dividida por ano. No primeiro ano, os autores coletaram dados sobre o uso do JFLAP em 7 universidades, com dados de 55 estudantes no começo do semestre, e de 33 estudantes ao final do semestre, além de realizarem uma pesquisa sobre a usabilidade do JFLAP. Eles constataram que a maior parte do uso do JFLAP por parte dos alunos foi para a resolução de atividades da disciplina cursada, fora do horário de aula. A pesquisa também mostrou que as opniões dos usuário foram positivas sobre a usabilidade e utilidade do JFLAP, além de apontarem para o fato dos alunos se sentirem mais motivados e confiantes nas suas habilidades após utilizar a aplicação.

Para o segundo ano de avaliação, os autores utilizaram os dados analisados no ano anterior para refinar os novos testes, realizando testes no começo e ao final do semestre. Neste

Figura 10 – Menu para seleção de funcionalidades no JFLAP.



Fonte: Elaborada pelo autor.

ano, uma maior quantidade de respostas foram obtidas em relação ao ano anterior, partindo de 6 universidades. Os novos resultados demonstraram que a maior parte dos estudantes gostaram da usabilidade do JFLAP e o acharam uma boa ferramenta. Os novos testes também mostraram que os estudantes participantes apresentaram um pouco menos de confiança nas suas habilidades, mas que acertaram mais questões no teste realizado ao final do semestre.

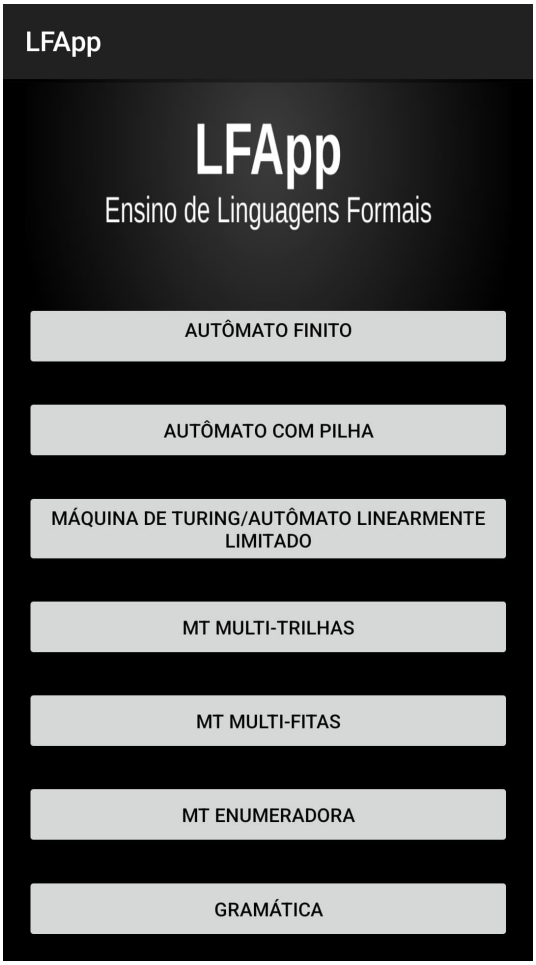
Rodger *et al.* (2009) possui uma grande relação com o trabalho proposto. Em ambos os trabalhos há apresentação de uma ferramenta que contribui de forma positiva para a prática e o aprendizado dos conceitos de MTs. Ambos os trabalhos também realizam a avaliação da aplicação com estudantes, utilizando testes e formulários. Ademais, diversas funcionalidades da ferramenta apresentada em Rodger *et al.* (2009), relacionadas ao estudo de MTs, foram aproveitadas por este trabalho, como descrito na Seção 4.4.

3.2 *A mobile app for teaching formal languages and automata*

Em Pereira e Terra (2018), foi apresentado e avaliado um aplicativo para dispositivos móveis *Android* que permite o estudo de vários conceitos de Teoria da Computação. O *Formal Languages and Automata Application* (FLApp) permite a criação e a simulação de diversos tipos de máquinas abstratas, dentre elas a MT e algumas de suas variantes, além de permitir a

conversão entre autômatos finitos determinísticos e não determinísticos, checagem de ambiguidades, remoção de recursões, minimização de autômatos, e muito mais. Na Figura 11 é possível observar o menu principal do aplicativo FLApp com a lista de todas as suas funcionalidades. Observe que o nome do aplicativo na figura é LFApp. Isso ocorre porque a versão disponibilizada pelos autores está em português do Brasil, com todos os textos, incluindo o título, traduzidos. No entanto, este trabalho continuará referindo-se ao aplicativo pelo seu nome em inglês.

Figura 11 – Menu para seleção de funcionalidades no FLApp.



Fonte: Elaborada pelo autor.

Os autores realizaram dois tipos de avaliação com a aplicação. A primeira avaliação foi focada em identificar possíveis erros ou problemas na aplicação, realizando testes unitários que abrangeram cerca de 90% da base de código, tornando o aplicativo bastante seguro em relação a comportamentos inesperados e erros. A segunda avaliação foi no contexto educacional, para constatar os impactos que o aplicativo traria para o aprendizado de um grupo de estudantes.

Para avaliar a eficácia no quesito educacional do aplicativo, os autores o integraram

em uma sala de aula da disciplina de Linguagens Formais e Autômatos. O aplicativo foi utilizado para ensino e criação de questões para atividades e provas por parte do professor, além de ter sido utilizado fortemente pelos estudantes para estudar, criar, e testar suas soluções para as atividades propostas. Por meio dessa integração, os autores notaram vários benefícios no aprendizado, motivação e confiança por parte dos estudantes. Esse fator também reduziu a quantidade de estudantes que dependiam de aulas extras para aprender a disciplina. Além disso, os autores também notaram que o FLApp reduz a dificuldade dos professores ao elaborar questões de Linguagens Formais e Autômatos, pois os professores podem criar as questões, testá-las e modificá-las, de forma muito mais simples e rápida.

Similar ao trabalho Pereira e Terra (2018), o presente trabalho apresenta e avalia uma aplicação que pode ser utilizada por meio de dispositivos móveis *Android*, embora a ferramenta apresentada por este trabalho possa ser utilizada em diversos outros dispositivos e sistemas operacionais. Outra similaridade é a forma de avaliação das ferramentas, onde ambos os trabalhos avaliaram suas aplicações por meio da utilização real de estudantes em sala de aula, que atribuiam *feedbacks* posteriores.

3.3 Ensinando Teoria da Computação com Jupyter Notebook

No trabalho Vasconcelos e Guerra (2023) foi apresentada e avaliada a Teocomp, uma biblioteca destinada ao auxílio no ensino da disciplina de Teoria da Computação. Ela foi construída utilizando a linguagem de programação *Python* e disponibilizada através de *notebooks* utilizando a ferramenta *Jupyter Notebook*. Um *notebook Jupyter* é um ambiente para desenvolvimento interativo de código.

Neste trabalho, os autores dividiram a diciplina de Teoria da Computação em módulos e destinaram um *notebook* para cada módulo. Cada *notebook* continha a descrição teórica do assunto estudado, além de uma série de exercícios executáveis e interativos. O segundo módulo da disciplina foi especialmente focado em MTs e suas variantes, com teoria e prática de seus conceitos, utilizando os *notebooks*. A Figura 12 mostra um exemplo de código que gera uma MT utilizando a biblioteca Teocomp.

Para avaliar as contribuições que a biblioteca teve no aprendizado dos estudantes, os autores realizaram uma pesquisa por meio de um formulário *on-line*, utilizando a ferramenta *Google Forms*, com estudantes da disciplina de Teoria da Computação nos níveis de graduação e mestrado, em dois semestres distintos. As respostas do formulário demonstraram que os

Figura 12 – MT descrita utilizando a biblioteca Teocomp.

```
Q = {'q0','q1','q2','q3','q4'}
Sigma = {'0','1'}
Gamma = {'0','1','X','Y','□'}
blank = '□'
delta = {('q0','0'):('q1','X','R'),
         ('q1','0'):('q1','0','R'),
         ('q1','Y'):('q1','Y','R'),
         ('q1','1'):('q2','Y','L'),
         ('q2','0'):('q2','0','L'),
         ('q2','Y'):('q2','Y','L'),
         ('q2','X'):('q0','X','R'),
         ('q0','Y'):('q3','Y','R'),
         ('q3','Y'):('q3','Y','R'),
         ('q3','□'):('q4','□','L')}
q0 = 'q0'
F  = {'q4'}
M = MT(Q,Sigma,Gamma,delta,q0,blank,F)
```

Fonte: Vasconcelos e Guerra (2023, p. 7).

estudantes utilizaram a biblioteca para resolver atividades síncronas e assíncronas, além de confirmar que os estudantes a utilizavam de forma bastante regular ao decorrer da semana. A avaliação também confirmou a contribuição da biblioteca para o entendimento de assuntos complexos de Teoria da Computação. Em geral, a Teocomp foi avaliada pela grande maioria dos estudantes como excelente ou muito boa.

O trabalho Vasconcelos e Guerra (2023) possui diversas semelhanças com o trabalho proposto. A principal semelhança entre ambos os trabalho é a apresentação e a avaliação de uma ferramenta para o entendimento e a prática dos conceitos de MTs. Além disso, a ferramenta desenvolvida por Vasconcelos e Guerra (2023) também pode ser utilizada na web por meio de *websites* como *Google Colab*[1], assim como a ferramenta proposta por este trabalho. Somado a isso, a realização de uma pesquisa para saber a satisfação dos estudantes e os efeitos em sua educação acontecem em ambos os trabalhos.

3.4 Análise comparativa dos trabalhos similares

O presente trabalho desenvolveu e avaliou uma ferramenta digital com foco principal na prática, no estudo e no ensino de MTs. Diferente do trabalho Vasconcelos e Guerra (2023), este trabalho propõe uma ferramenta que conta com uma interface gráfica que possibilita uma

[1] https://colab.research.google.com/

interação mais natural, sem que haja necessidade de interação com códigos ou linguagens de programação de forma direta pelo usuário.

De forma semelhante aos trabalhos de Rodger *et al.* (2009), Pereira e Terra (2018), Vasconcelos e Guerra (2023), a ferramenta desenvolvida por este trabalho visa possibilitar o estudo da MT clássica e diversas variantes de MT, por meio da sua gama de funcionalidades. Além disso, neste trabalho também será realizada uma avaliação sobre a contribuição do *website* para a aquisição de conhecimento pelos estudantes, assim como feito nos trabalhos relacionados. O Quadro 1 compara este trabalho aos demais apresentados.

Quadro 1 – Comparativo entre os trabalhos relacionados e este trabalho.

	Rodger *et al.* (2009)	Pereira e Terra (2018)	Vasconcelos e Guerra (2023)	Este trabalho
Há o desenvolvimento/apresentação de uma ferramenta	Sim	Sim	Sim	Sim
Desenvolveu uma ferramenta que possibilita o estudo de MTs	Sim	Sim	Sim	Sim
Desenvolveu uma ferramenta com foco em boas práticas de UX e UI	Não	Não	Não	Sim
Desenvolveu uma ferramenta multiplataforma	Não	Não	Sim	Sim
Avalia a ferramenta desenvolvida	Sim	Sim	Sim	Sim

Fonte: Elaborado pelo autor.

4 FERRAMENTAS RELACIONADAS

Este capítulo descreve com mais detalhes as ferramentas similares apresentadas pelos trabalhos relacionados descritos no Capítulo 3. Estas descrições serão feitas com base nas experiências pessoais do autor relacionadas ao uso e testes com as ferramentas descritas. O foco principal deste capítulo é entender as utilidades e funcionalidades das ferramentas principalmente para tópicos relacionados a MTs no contexto educacional.

Ao final, uma breve comparação das ferramentas também é realizada, sendo este o primeiro passo para o desenvolvimento da aplicação proposta para este trabalho, visto que o mesmo utiliza as ferramentas similares para extrair as principais funcionalidades e características que este tipo de ferramenta deve possuir.

4.1 JFLAP

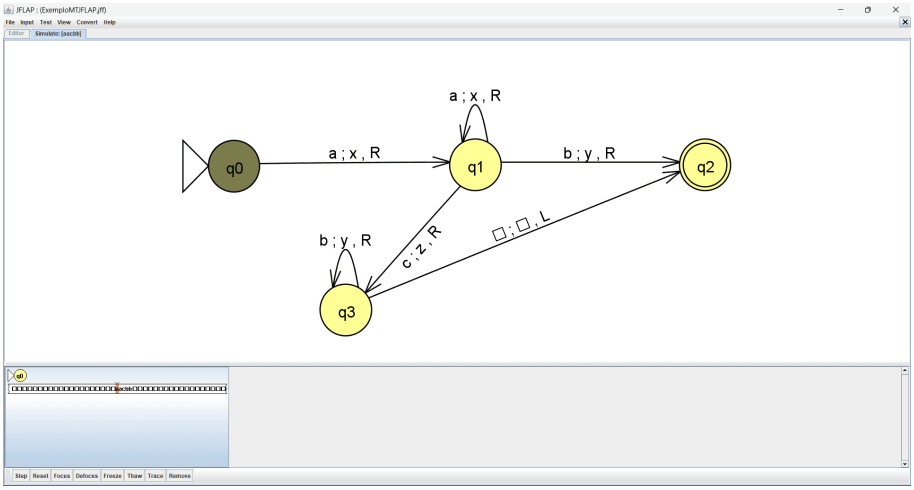
O *Java Formal Languages and Automata Package* (JFLAP) é um *software desktop* que permite a criação interativa de diversas estruturas abstratas de Teoria da Computação. Voltado para MTs, o JFLAP apresenta uma gama de funcionalidades que permitem importar, exportar, criar, visualizar, e testar a MT clássica e diversas outras variantes de MTs, dentre elas: MTs não determinísticas e MTs multifitas com até 5 fitas. Ele também permite a criação e integração de sub-rotinas (integração de MTs dentro de outras MTs), chamadas de blocos de construção (do inglês, *building blocks*). Para mais detalhes sobre o JFLAP, consultar (Procopiuc *et al.*, 1996).

O JFLAP é um *software desktop* instalável em sistemas operacionais *Windows*, *MacOS*, *Linux/Unix*. Ele possui uma interface gráfica que permite uma interação natural com a ferramenta, sem necessidade de escrita de códigos. O JFLAP é organizado por meio de menus que estruturam a sua navegação, facilitando-a em muitos casos.

No contexto da criação e testes de MTs, o JFLAP permite a criação interativa das MTs por meio do movimento de arrasta e solta ao selecionar os botões correspondentes com as ações desejadas na aplicação. Além disso, o JFLAP permite visualizar as MTs e sua computação por meio de diagramas de estados e também de sua fita, com diversos controles de simulação, como por exemplo, a simulação passo a passo, onde o usuário pode aprender um pouco mais ao ver as configurações que a fita necessitou passar para retornar sua resposta. Veja um exemplo de uma MT construída utilizando o JFLAP na Figura 13.

Somado a isso, o JFLAP permite a observação do não determinismo ao mostrar a

Figura 13 – MT construída utilizando o JFLAP.



Fonte: Elaborada pelo autor.

configuração da fita da MT nas várias derivações possíveis da computação. Ele também permite a observação das configurações das multifitas de uma MT multifitas a cada passo da computação, podendo até mesmo haver a combinação de multifitas e não determinismo.

No geral, o JFLAP possui diversos pontos fortes que inclusive foram considerados para a construção da Turing Station, tendo diversas funcionalidades e formas de realizar a computação utilizadas como base. Os pontos mais fortes do JFLAP são a sua interatividade, que proporciona um formato bem agradável de criação, visualização e testes das MTs, e também a possibilidade de praticar diversas variantes de MTs.

Já um dos fatores negativos observados no JFLAP é o fato da ferramenta estar limitada a utilização por meio de um dispositivo *desktop*, o que impossibilita sua utilização multiplataforma. A necessidade da instalação em um computador de mesa dificulta o uso da ferramenta dentro da sala de aula, pois muitas vezes este tipo de equipamento não está disponível para todos os alunos no momento da aula.

Ademais, o JFLAP possui certos problemas de usabilidade, como notado na avaliação de experiência de uso realizada por Silva *et al.* (2023). Esses problemas de usabilidade provavelmente estão ligados ao fato de sua interface não ter se atualizado ao longo dos anos, se mantendo praticamente a mesma desde a sua criação.

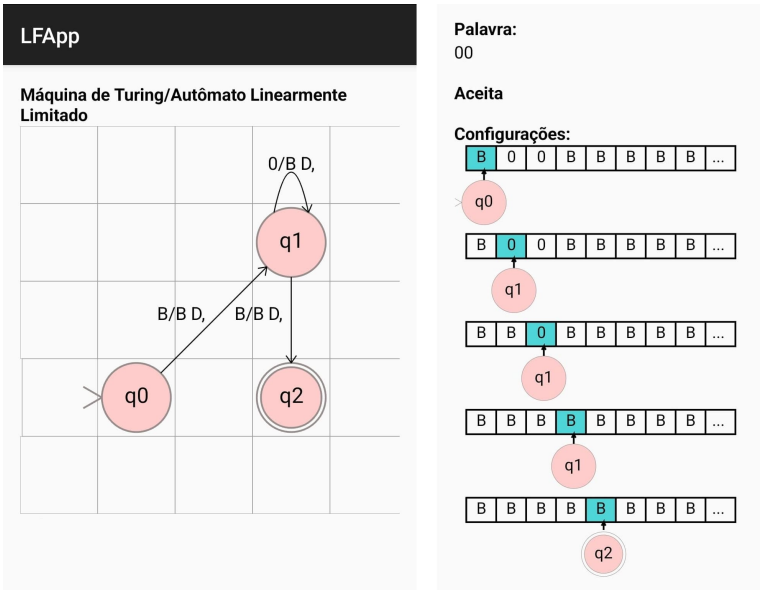
4.2 FLApp

O *Formal Languages and Automata Application* (FLApp) é um aplicativo desenvolvido para dispositivos móveis com sistema operacional *Android*, que permite a criação interativa

de diversas estruturas abstratas de Teoria da Computação (Pereira; Terra, 2018). Ele também possui uma interface gráfica que permite uma interação natural com a ferramenta sem necessidade de escrita de códigos. O FLApp é organizado por meio de um menu que estrutura a sua navegação e escolha de funcionalidades. A Figura 11 apresentada na Seção 3.2 mostra o menu principal do aplicativo.

Relacionado a MTs, o aplicativo permite a criação e simulação de MTs determinísticas e algumas variantes, como MT multifitas, de forma interativa por meio de toques na tela. A MT é criada em formato de diagrama de estados e o resultado da computação de uma palavra é mostrado logo abaixo, apresentando todas as configurações da fita nas etapas da computação, mas sem controle granular sobre a simulação. Na Figura 14 é possível observar, ao lado esquerdo, um exemplo de diagrama de estados de uma MT que testa se a palavra é composta apenas por "0"s, enquanto ao lado direito estão as configurações da computação da palavra "00" sobre esta MT, dentro do aplicativo FLApp.

Figura 14 – Simulação de MT no aplicativo FLApp.



Fonte: Elaborada pelo autor.

Um dos principais pontos fortes do FLApp é que ele permite o estudo e a prática da criação das estruturas de Teoria da Computação por meio de dispositivos móveis. Esta característica se alinha com os tempos modernos onde a maioria dos estudantes universitários possuem dispositivos móveis consigo enquanto estão em sala de aula.

Um dos principais fatores negativos sobre o FLApp está relacionado a sua usabilidade. Houve dificuldades para criar as MTs pois os controles de criação não pareciam tão claros, além

de apresentarem diversos erros. Isso também está relacionado ao fato de que o FLApp não possui nenhum tipo de documentação, ficando a cargo do usuário compreender totalmente como utilizá-lo. Outro ponto negativo se deve ao fato de que o FLApp, no momento em que foi utilizado para esta análise, encontrava-se indisponível na loja de aplicativos para sistemas *Android*, a *Play Store*. Sua utilização só foi possível por meio de um processo mais complexo de instalação, utilizando o repositório de códigos que os autores disponibilizaram no trabalho descrito em 3.2, dificultando a sua utilização por usuários comuns.

4.3 Teocomp

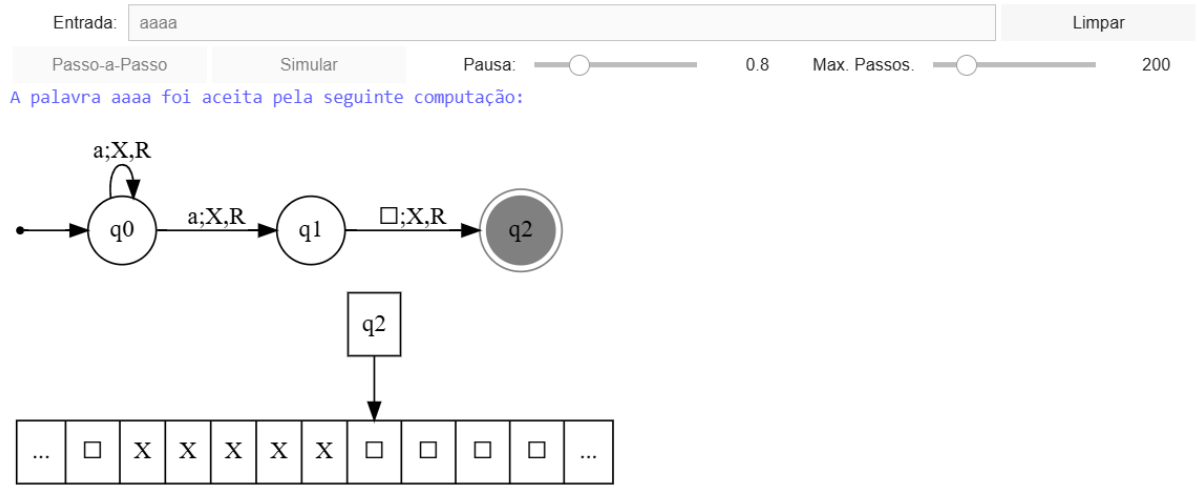
A Teocomp é uma biblioteca que pode ser utilizada para o estudo e ensino de MTs, variantes de MTs, e outros conceitos de Teoria da Computação, tendo sido desenvolvida com a linguagem de programação *Python* (Vasconcelos; Guerra, 2023). A Teocomp não possui uma interface gráfica nativa que permita uma interação tão natural quanto as outras ferramentas apresentadas. Para utilizá-la, os estudantes descrevem as máquinas em formato de código, na forma que a biblioteca espera receber, e recebem como retorno a visualização da máquina desejada, representada por meio de um diagrama de estados e/ou fita.

A biblioteca também permite o teste de palavras nas MTs desenvolvidas. Para testar, os estudantes devem listar as palavras em uma lista *Python* e executar uma função programada pelos autores que retornará como resposta *False*, se a palavra não pertencer a linguagem que a MT espera, ou *True*, caso pertença, para cada uma das palavras listadas. A biblioteca também permite a realização de teste de palavras por meio de uma pequena interface gerada por uma biblioteca de gráficos integrada, com isso é possível a utilização de controles de teste, permitindo aos estudantes observarem os passos da computação até a sua finalização. A Figura 15 mostra o teste da palavra "aaaa" na biblioteca Teocomp, em uma MTND que aceita palavras pertencentes a linguagem $A = \{aa^*\}$, ou seja, palavras com um ou mais "a"s.

Para a utilização da biblioteca Teocomp, os usuários não necessitam ter conhecimento aprofundado em *Python*, no entanto, os usuários precisam se adaptar ao formato de entrada que a biblioteca e a linguagem *Python* esperam receber, para que não hajam erros na execução das máquinas descritas.

Os *notebooks* utilizados em conjunto com a biblioteca, como mostrado na Seção 3.3, contribuem para uma boa utilização e aquisição de conhecimento por parte de seus usuários, sendo seu teor didático um dos pontos mais positivos da mesma.

Figura 15 – Teste de palavra em uma MTND utilizando a biblioteca Teocomp.



Fonte: Elaborada pelo autor.

Um ponto negativo sobre a Teocomp se deve principalmente pela biblioteca de Vasconcelos e Guerra (2023) não possuir uma interface gráfica interativa para o *design* das MTs, devendo ser utilizada primordialmente por meio de código escrito. Esta característica pode afastar diversos usuários que não possuem afinidade com a escrita de código na linguagem *Python*, embora saber a linguagem de forma aprofudanda não seja um pré-requisito para a sua utilização.

4.4 Análise comparativa das ferramentas similares

A análise e comparação das ferramentas similares influenciou diretamente no desenvolvimento da aplicação proposta. Essa análise teve como um de seus objetivos elencar as principais características e funcionalidades destas plataformas, de forma mais detalhada, para que as melhores funcionalidades e qualidades pudessem ser incorporadas a aplicação Turing Station. Além disso, baseado em uma análise empírica, foi avaliado também aspectos de UX e UI dessas ferramentas que pudessem contribuir ou que devessem ser evitados na Turing Station.

Para melhor organização, o Quadro 2 faz um comparativo das informações relacionadas as principais características e funcionalidades com foco em MTs entre as ferramentas similares. Já o Quadro 3 faz um comparativo dos principais aspectos notados pelo autor com relação a UX e UI entre as ferramentas similares. As principais características descrevem brevemente o escopo global da ferramenta, como plataforma computacional na qual pode-se usar a ferramenta, modo de uso, etc. Já quando se trata das principais funcionalidades com foco em MTs, visa-se resumir as informações específicas sobre MTs, visto que dentre as pla-

taformas selecionadas há aquelas que abrangem uma série de outros assuntos relacionados a Teoria da Computação. Ao se tratar de UX e UI, destacam-se os aspectos positivos ou negativos relacionados a usabilidade da aplicação.

Para o preenchimento dos quadros comparativos, foi utilizada uma escala descritiva qualitativa para definir os valores de comparação. A seguir estão listados os valores da escala e suas respectivas descrições:

- **Excelente:** Indica que a característica ou funcionalidade é plenamente atendida, apresentando alta qualidade e eficiência.
- **Boa:** Representa que a característica ou funcionalidade é bem atendida, com desempenho satisfatório, mas com possibilidade de melhorias.
- **Média:** Indica que a característica ou funcionalidade está em um nível aceitável, mas com limitações perceptíveis que podem impactar a experiência.
- **Fraca:** Representa que a característica ou funcionalidade é inadequada ou deficiente, apresentando problemas que comprometem sua utilização.
- **Inexistente:** Indica a completa ausência da característica ou funcionalidade, sem suporte ou implementação disponível.

Quadro 2 – Comparativo das características gerais e funcionalidades relacionadas a MT.

Características	**JFLAP**	**FLApp**	**TEOCOMP**
Facilidade de instalação	Média	Fraca	Média
Suporte a criação, visualização e testes de MT	Excelente	Média	Excelente
Suporte a criação, visualização e testes de variantes de MT	Excelente	Média	Excelente
Qualidade da visualização do diagrama de estados da MT	Excelente	Boa	Excelente
Qualidade da visualização da fita da MT	Excelente	Excelente	Excelente
Visualização do passo a passo da computação	Excelente	Excelente	Excelente
Qualidade dos controles de simulação	Excelente	Inexistente	Excelente
Ajuda e documentação	Média	Inexistente	Média
Suporte a multiplataformas	Média	Inexistente	Boa
Suporte a múltiplos testes de palavras simultaneamente (multiteste)	Excelente	Inexistente	Excelente
Suporte a utilização de sub-rotinas	Excelente	Inexistente	Excelente
Suporte a importação de arquivo de MT	Excelente	Inexistente	Inexistente
Suporte a exportação de arquivo de MT	Excelente	Inexistente	Inexistente

Fonte: Elaborado pelo autor.

Como é possível visualizar no Quadro 2, as três ferramentas foram construídas para serem executadas em plataformas computacionais bem restritas, possuindo pouco ou nenhum suporte para múltiplas plataformas, bem como suas formas de interação também se diferem, possuindo interface gráfica interativa ou não. Essas características influenciam a quantidade

Quadro 3 – Comparativo das características relacionadas a UX e UI.

Características	**JFLAP**	**FLApp**	**TEOCOMP**
Organização da navegação	Boa	Boa	Fraca
Intuitividade da navegação	Média	Média	Fraca
Adaptabilidade em diferentes tamanhos de dispositivos (Responsividade)	Fraca	Fraca	Média
Reatibilidade das ações	Boa	Boa	Média
Compatibilidade com dispositivos *desktops*	Excelente	Fraca	Excelente
Compatibilidade com dispositivos móveis	Fraca	Excelente	Média
Qualidade do suporte e ajuda dentro da aplicação (dicas, alertas, notificações)	Fraca	Fraca	Fraca
Quantidade e qualidade dos atalhos de teclado	Média	Fraca	Média
Facilidade na criação de MTs por meio de interface gráfica	Excelente	Média	Inexistente
Performance	Excelente	Excelente	Excelente
Curva de aprendizado	Média	Média	Média
Qualidade da interface do usuário	Boa	Média	Fraca
Qualidade e intuitividade dos ícones	Média	Média	Média
Qualidade da paleta de cores	Média	Média	Média

Fonte: Elaborado pelo autor.

de usuários que tais ferramentas podem abranger, pois os usuários podem estar limitados a utilizar tais ferramentas apenas por um tipo de plataforma computacional específica, com poucas exceções. Somado a isso, algumas das ferramentas apresentam a necessidade de instalar outras dependências para que possam ser utilizadas, dificultando ainda mais a sua utilização.

Quando se trata das principais funcionalidades relacionadas ao estudo de MTs, o JFLAP se destaca dentre as outras ferramentas devido a alta quantidade e a qualidade das funções que ele oferece para criação, edição e simulação de MTs. Ele permite tudo isso de forma interativa por meio de sua interface gráfica. Além disso, o JFLAP oferece a possibilidade de importação e exportação de arquivos para utilização posterior, bem como a integração das MTs com sub-rotinas, que se resume a utilização de uma MT dentro de outra MT como uma parte integrante.

A biblioteca de Vasconcelos e Guerra (2023), possui praticamente as mesmas funções do JFLAP quando relacionadas a MTs, diferenciando-se negativamente por não possibilitar a importação e exportação de MTs em formato de arquivo, devido ao fato da sua utilização ser feita por meio de códigos.

Nesta análise foi possível notar que quando relacionado ao estudo de MTs o FLApp possuía apenas as funcionalidades mais básicas para esta função. Nele não estavam presentes diversas variantes de MT importantes, além de não possibilitar a exportação e a importação de MTs, e também a realização de multitestes. Todos esses fatores contribuem para que sua

utilização possa ser considerada limitada em termos de aprendizado, visto que estas funcionalidades mais avançadas são de extrema importância na aprendizagem dos conceitos de MTs pelos estudantes.

Com respeito aos aspectos de UX apresentados no Quadro 3, foi possível notar que tais ferramentas possuem diversos pontos positivos e negativos. Algumas delas possuem menus para navegação que organizam as funções da ferramenta, outras não. Ainda sobre navegação, todas as ferramentas testadas apresentam sérios problemas de estrutura e desordem visual, dificultando a localização de informações importante devido a falta de estrutura lógica e intuitiva. A falta de compatibilidade com dispositivos móveis também é um grande problema para o JFLAP e a biblioteca Teocomp.

Ademais, foi possível notar que algumas respostas das ferramentas para as ações do usuário não são tão claras, ou ficam ocultas, dificultando a detecção de erros. Outros problemas de UX são detectados quando se analisa a área de suporte e ajuda, que além de estar em falta ou ser bem antiga, não está organizada de forma intuitiva. Além disso, a falta de atalhos de teclados, a alta curva de aprendizado e os *feedbacks* que muitas vezes não são mostrados de forma adequada, tornam o processo de utilização de tais ferramentas um pouco dificultoso.

Em termos de UI, também é possível notar no Quadro 3 diversos pontos a serem melhorados, como a interface não padronizada e ícones não tão intuitivos. Além disso, as paletas de cores escolhidas pelo JFLAP e o FLApp são formadas majoritariamente pela cor branca e tons claros, o que pode causar fadiga visual aos usuários que não podem se expor ao brilho forte de uma tela por muito tempo, por exemplo.

Em geral, cada uma das ferramentas de ensino analisadas contribuiu de diferentes maneiras para este trabalho, principalmente com os fatores que mais as destacam. O aplicativo FLApp, mostrou que é possível levar o ensino de MTs para dispositivos móveis. A biblioteca Teocomp de Vasconcelos e Guerra (2023) mostrou que a web é um ótimo meio de ensino quando utilizada de forma correta, levando o ensino a uma maior quantidade de usuários e dispositivos. Já o JFLAP, por ser o mais completo em termos de funcionalidades, contribuiu principalmente para a definição dos requisitos gerais da Turing Station, além de ter ajudado a definir aspectos relacionados a estrutura e formato da simulação das diferentes variantes de MTs.

5 DESENVOLVIMENTO DA APLICAÇÃO

Este capítulo detalha as etapas seguidas para a construção da aplicação proposta. A primeira etapa, referente à análise de ferramentas similares, já foi abordada na Seção 4.4. Assim, este capítulo se concentrará nas etapas subsequentes, conforme ilustrado na Figura 16.

Figura 16 – Etapas seguidas para a construção da aplicação.

Fonte: Elaborada pelo autor.

5.1 Definição dos requisitos

Nesta etapa foram definidos os requisitos de sistema necessários para a aplicação, que pode ser dividida em simulador (área de criação e testes de MTs) e *website* (aplicação completa). Os requisitos se referem as funcionalidades, serviços e restrições do sistema, que servirão para suprir as necessidades de determinado usuário/cliente. Existem diferentes tipos e níveis de requisitos, definidos baseados em seu nível de detalhamento e público alvo (Sommerville, 2015).

Neste trabalho, utilizaremos dois tipos de requisitos para definirmos as especificações da aplicação a ser desenvolvida, sendo eles: Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF). Isso se deve ao fato de já se conhecer os problemas mais comuns entre os usuários deste tipo de aplicação, devido a análise e utilização de aplicações similares, além do contexto da aplicação ser mais restrito de que uma aplicação de maior porte, como por exemplo o próprio JFLAP.

Segundo Sommerville (2015), os RFs podem ser entendidos como um tipo de requisito que foca em descrever os serviços que o sistema deve prestar, bem como definir como o sistema deve funcionar, agir em determinados cenários, se comportar com determinadas entradas, e em casos específicos, explicar o que o sistema não deve fazer.

Já os RNFs se referem as restrições impostas sobre o sistema ou parte dele. Eles não definem um serviço ou ação do sistema, mas impõe regras para seu funcionamento, como tempo de resposta a uma requisição, usabilidade, segurança, entre outros. Eles funcionam como critérios de julgamento para o funcionamento do sistema.

No Quadro 4 estão dispostos os RFs e os RNFs do simulador de MTs. Já no Quadro

5 estão dispostos os RFs e os RNFs gerais da Turing Station. Esses quadros estão organizadas em duas colunas que definem o código do requisito e sua descrição, respectivamente. Observe que os códigos dos requisitos possuem RF como prefixo, quando se referem a requisitos funcionais, e RNF quando se referem a requisitos não funcionais.

Quadro 4 – Requisitos do simulador de MT.

Requisito	**Descrição**
RF01a	Permitir que o usuário desenvolva o *design* de MT e suas variantes de forma gráfica e interativa por meio de toques, cliques, e teclas de atalho.
RF02a	Permitir que o usuário desenvolva MTs clássicas e suas variantes, como a MT com fita finita à esquerda, com opção de parada, multifitas e não determinísticas. O simulador deve manter todas as suas funcionalidades para cada uma das variantes de MT.
RF03a	Permitir que o usuário adicione, selecione, mova, e remova estados em uma MT, além de possibilitar a definição de estado inicial e a definição de múltiplos estados finais.
RF04a	Permitir que o usuário adicione, selecione, edite e remova regras de transição.
RF05a	Permitir que o usuário exporte uma MT como arquivo de imagem.
RF06a	Permitir que o usuário exporte uma MT como arquivo *JavaScript Object Notation* (JSON).
RF07a	Permitir que o usuário importe um arquivo de MT em formato JSON, compatível com a aplicação, possibilitando a sua reutilização.
RF08a	Permitir que o usuário realize diversas ações dentro do simulador, como: fazer e desfazer ações, mover área de *design*, limpar toda a área de *design*, aumentar ou diminuir o tamanho das MTs (*zoom in/zoom out*).
RF09a	Permitir que o usuário teste palavras com as MTs desenvolvidas, obtendo controle total da computação, podendo avançar, retroceder e automatizar o teste da palavra.
RF10a	Permitir que o usuário teste palvras com as MTs de forma intuitiva e visual, podendo observar a simulação por meio de diagrama de estados e das configurações da fita da MT.
RF11a	Permitir que o usuário teste múltiplas palavras simultaneamente com a MT criada.
RF12a	Permitir que o usuário obtenha retorno visual em caso de erros, alertas, e sucesso.
RF13a	Permitir que o usuário, por meio de mensagens informativas habilitáveis, receba o conhecimento necessário para usar cada funcionalidade do simulador.
RF14a	Permitir que o usuário renomeie um simulador.
RF15a	Permitir que o usuário duplique um simulador (adicionando a cópia na lista de simuladores).
RF16a	Permitir que o usuário apague um simulador (removendo-o da lista de simuladores).
RF17a	Permitir que o usuário utilize o simulador em tela cheia.
RNF01a	O simulador deve ser capaz de responder aos comandos do usuário de forma assertiva e em tempo hábil.
RNF02a	O simulador deve se adaptar bem a tamanhos de telas diferentes, mantendo todas as suas funcionalidades e facilidades na sua utilização.
RNF03a	O código do simulador deve ser de fácil manutenção, facilitando a correção de erros e atualizações pelo desenvolvedor.

Fonte: Elaborado pelo autor.

Como é possível observar por meio do Quadro 4, a maioria dos RFs do simulador foram definidos baseados nas ferramentas similares analisadas por este trabalho. Vários requisitos extras também foram adicionados, como as mensagens de ajuda e *feedbacks*, atalhos de teclado, entre outros. Essas novas funcionalidades visam sanar alguns problemas de usabilidade que as ferramentas similares apresentaram. Além disso, foram definidos RNFs que buscam guiar o desenvolvimento da aplicação por um caminho que facilite uma futura melhoria das

funcionalidades e correção de erros. Também buscou-se esclarecer que a aplicação deve, desde seu princípio, ser adaptável para diversos dispositivos diferentes, e rápida o suficiente para atender as solicitações dos usuários em tempo hábil.

Quadro 5 – Requisitos gerais da Turing Station.

Requisito	Descrição
RF01b	A aplicação deve dispor de uma página denominada "Meus simuladores", onde é possível ver a lista de simuladores (MTs) criados, adicionar mais simuladores, filtrar e ordenar os simuladores visíveis, além de possibilitar a exportação e importação de diversas MTs em formato JSON de forma simultânea.
RF02b	A aplicação deve dispor de uma página denominada "Instruções de uso", onde o usuário pode aprender de forma mais aprofundada como utilizar todas as funcionalidades da própria aplicação, com tutoriais em video e/ou artigos de texto.
RF03b	A aplicação deve dispor de uma página denominada "Teste de conhecimento", onde o usuário pode resolver desafios relacionados a MTs utilizando o simulador, e receber *feedbacks* apropriados, aprimorando seus conhecimentos.
RNF01b	A aplicação deve ser capaz de responder aos comandos do usuário de forma assertiva e em tempo hábil.
RNF02b	A aplicação deve se adaptar bem a tamanhos de telas diferentes, mantendo todas as suas funcionalidades e facilidades na sua utilização.
RNF03b	O código da aplicação deve ser de fácil manutenção, facilitando a correção de erros e atualizações pelo desenvolvedor.
RNF04b	A aplicação deve poder ser utilizada de forma *on-line* por meio da web e também de forma *off-line* por meio da instalação local.

Fonte: Elaborado pelo autor.

Observando o Quadro 5, percebe-se que foram descritos RFs que definem as páginas da Turing Station. Definiu-se que a Turing Station precisa disponibilizar uma página que permite a visualização e realização de ações com todos os simuladores de MTs criados pelo usuário, disponibilizar uma página com instruções de uso da aplicação, além de dispor de uma página para que os usuários resolvam desafios e testem seus conhecimentos relacionados a MTs, o que pode motivá-los a aprender mais por meio da resoluções dos desafios propostos. Os RNFs gerais da Turing Station são equivalentes aos apresentados para o simulador de MTs no Quadro 4, com exceção do RNF04b, que fala sobre as formas de utilização da aplicação.

5.2 Prototipação visual

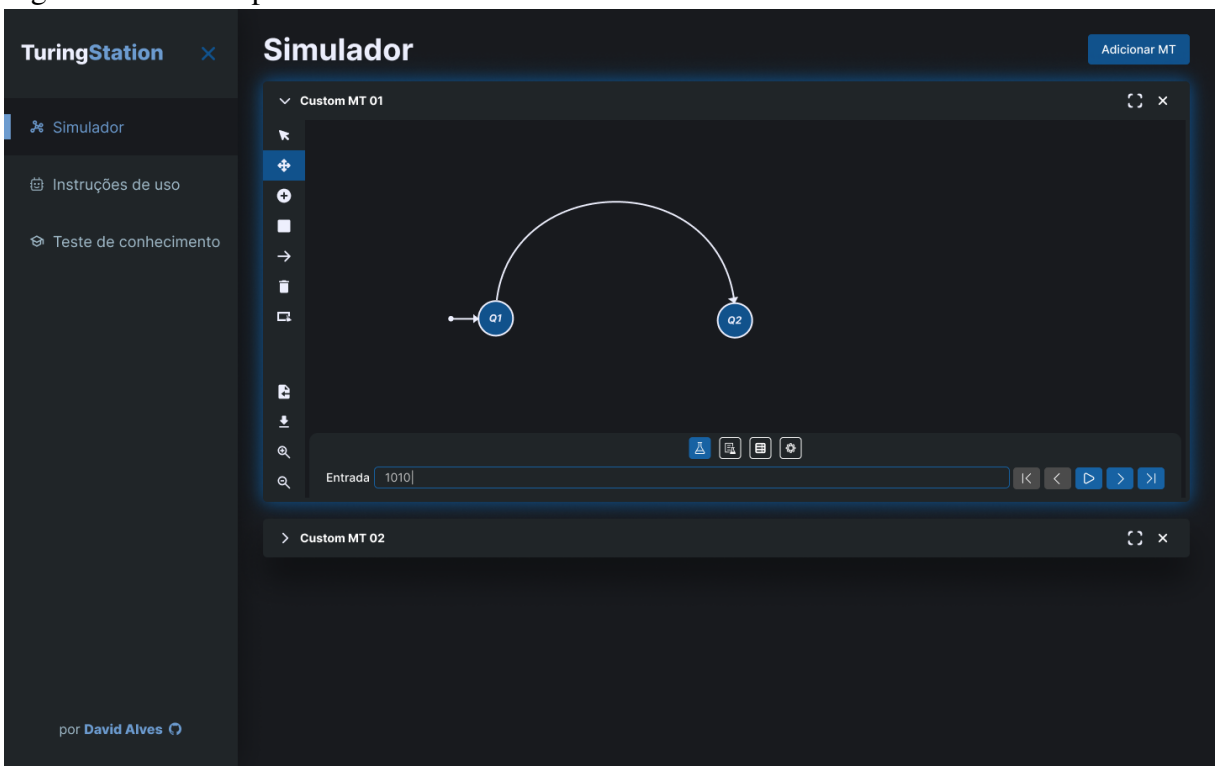
Nesta etapa foi realizado o desenvolvimento da prototipação visual da aplicação Turing Station. A prototipação buscou desenvolver um material capaz de apresentar toda a aparência do *website* e do simulador, antes mesmo de escrever uma única linha de código. Esta capacidade de apresentação auxilia no desenvolvimento do código da aplicação pois permite ao desenvolvedor saber como deve organizar seu código, definindo aspectos estilísticos e estruturais

de forma antecipada.

A prototipação de uma aplicação é a construção de um material que simule uma ou mais partes da aplicação real, sendo útil para apresentação de certos recursos ao cliente e recebimento de *feedbacks* antecipados (Budde *et al.*, 1990). A prototipação de toda a aplicação foi feita utilizando a ferramenta Figma[1], um editor gráfico utilizado para criação, compartilhamento e testes de *designs* para *softwares* e produtos digitais.

Baseado nos requisitos listados nos quadros 4 e 5, foi possível desenvolver um protótipo que mostrou de antemão que a aplicação iria, ao mesmo tempo que realiza as funcionalidades propostas, respeitar a maioria das boas práticas de UX e UI, buscando evitar alguns dos problemas observados nas ferramentas similares analisadas. A Figura 17 exibe uma versão preliminar do protótipo da tela "Meus Simuladores" na Turing Station.

Figura 17 – Protótipo da tela "Meus Simuladores".



Fonte: Elaborada pelo autor.

5.3 Desenvolvimento em código

Nesta etapa, foi realizada a seleção das tecnologias envolvidas no desenvolvimento da aplicação, bem como o desenvolvimento de código da mesma. A plataforma é acessada via

[1] https://www.figma.com/

navegadores web e por isso necessitou de utilização de tecnologias apropriadas para esse meio. Buscou-se no desenvolvimento da mesma a escalabilidade, ou seja, tornar possível a integração de novas funcionalidades sempre que necessário, além de facilitar a manutenção e correção de erros. Abaixo segue a lista das principais tecnologias utilizadas para a crição da Turing Station:

- HTML: atualmente em sua quinta versão, o *Hypertext Markup Language* (HTML) é a linguagem padrão para o desenvolvimento da estrutura de aplicações web. Ele foi inicialmente proposto com foco totalmente na estruturação das páginas, ficando a sua estilização a cargo dos *softwares* de renderização que o utilizam (Silva, 2019);
- CSS: o *Cascading Style Sheets* (CSS) surgiu para suprir uma das maiores necessidades do HTML, a possibilidade de estilizar uma página web. Com o CSS é possível estilizar textos, imagens, tabelas, layouts, praticamente todos os elementos HTML, além de tornar possível a criação de animações visuais com esses elementos (Meyer; Weyl, 2017);
- Tailwind CSS: o *Tailwind CSS* é um *framework* CSS que visa facilitar, acelerar e tornar mais escalável a estilização de *websites*. Ele é composto por classes de utilidades com funções CSS embutidas (Bhat, 2023);
- P5.js: o P5.js é uma biblioteca que possibilita a criação de códigos criativos, como animações e obras de arte. Ele atua no chamado *canvas*, uma *tag* HTML que possibilita a criação de elementos gráficos, como círculos, quadrados, imagens, animações, texto, e muito mais, utilizando a linguagem de programação *JavaScript*. O P5.js cria uma camada de abstração que facilita a utilização do *canvas* HTML, contribuindo principalmente para a agilidade e na diminuição da curva de aprendizagem (McCarthy *et al.*, 2015);
- JavaScript: o *JavaScript* é uma linguagem de programação criada com o propósito de dar interatividade as páginas web, desde ações mais básicas como o movimento de uma imagem, até mesmo o processamento de um formulário HTML (Silva, 2010);
- React: o *React* é uma biblioteca *JavaScript* voltada para a construção de interfaces de usuário (UI). Ele permite a criação de componentes reutilizáveis que refletem o estado da aplicação, promovendo maior modularidade e eficiência, além de acelerar o desenvolvimento de interfaces complexas. Com o *React*, é possível construir aplicações web dinâmicas e interativas de forma declarativa, utilizando o conceito de DOM virtual para otimizar o desempenho na atualização de elementos na interface (Gackenheimer, 2015).
- Node.js: o Node.js ou Node, como é conhecido, é um ambiente de execução *JavaScript* que atua no lado do servidor. O Node facilita o processamento concorrente em programas

> nativos web e torna possível executar código *JavaScript* em vários tipos de dispositivos. O ecossistema do Node permite a criação de servidores, linhas de comando, *web apps*, *scripts* e muito mais (Pereira, 2014).

Um dos principais objetivos ao desenvolver esta aplicação foi garantir que ela fosse leve e responsiva. Graças as tecnologias escolhidas, a aplicação é leve em termos de desempenho e pode ser executada diretamente em navegadores web sem a necessidade de instalações adicionais. Devido a sua interface responsiva, ela pode ser utilizada em uma variedade de dispositivos, incluindo *smartphones* e *tablets*. Isso amplia significativamente o acesso, permitindo que estudantes utilizem a ferramenta em sala de aula por meio de seus próprios dispositivos.

Algumas outras tecnologias auxiliares foram utilizadas para tornar a criação da aplicação mais rápida e escalável. O editor de texto Visual Studio Code[2] foi utilizado para a escrita de código. Ele possui uma série de *plug-ins* (funcionalidades adicionais) que facilitaram a detecção de erros no código, além de possuir uma boa quantidade de atalhos para acelerar o desenvolvimento. Outra tecnologia utilizada que foi essêncial para o desenvolvimento da aplicação foi o *Git*[3] utilizado para controle de código. Ele permite a criação de pontos de histórico de código, possibilitando o retrocesso em caso de erros ou necessidade. Juntamente ao *Git*, foi utilizado também o *GitHub*[4] que permite a hospedagem de código de forma online utilizando o *Git*. Esta hospedagem permite o compartilhamento de código entre desenvolvedores, facilitando a colaboração e aprimoramento das aplicações criadas. A Turing Station possui código aberto hospedado no *GitHub*, ou seja, qualquer pessoa pode contribuir para aprimorar as funcionalidades disponibilizadas nesta aplicação. O código-fonte da Turing Station está disponível em: https://github.com/dev-david-alves/turing-station.

[2] https://code.visualstudio.com/
[3] https://git-scm.com/
[4] https://github.com/

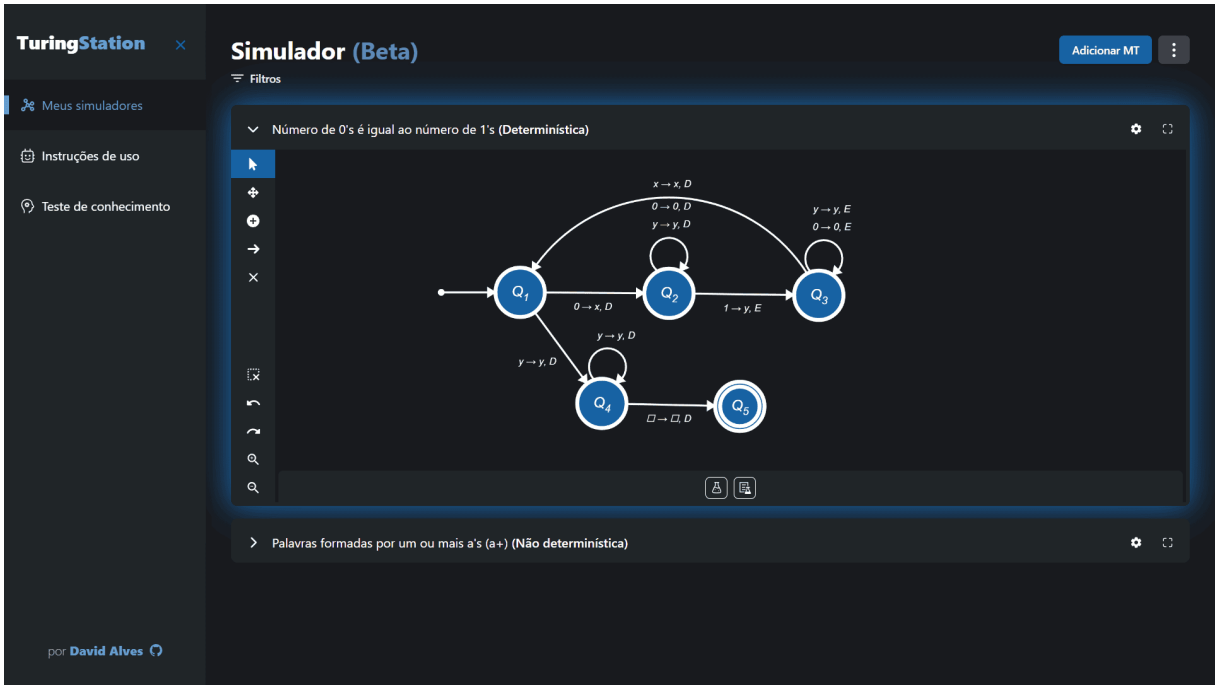
6 TURING STATION

Este capítulo apresenta a aplicação Turing Station, desenvolvida e avaliada por este trabalho. As próximas seções apresentam suas principais telas, características, e funcionalidades. É possível acessar a aplicação de forma *on-line* disponível em: https://dev-david-alves.github.io/turing-station.

6.1 Página Meus Simuladores

A página **Meus simuladores** é a tela inicial da aplicação. Nesta tela o usuário tem acesso a todas as funcionalidades de criação e execução de Máquinas de Turing (MT). Assim como em todas as subsequentes telas do *website*, ela conta com um menu lateral para navegação entre as páginas do *website*, que pode ser minimizado para ganho de área útil. Esta tela também conta com uma área principal onde o usuário pode ver e editar todas as suas máquinas criadas, selecionando com um clique alguma delas. A Figura 18 mostra a página **Meus simuladores** de maneira completa.

Figura 18 – Página Meus Simuladores.

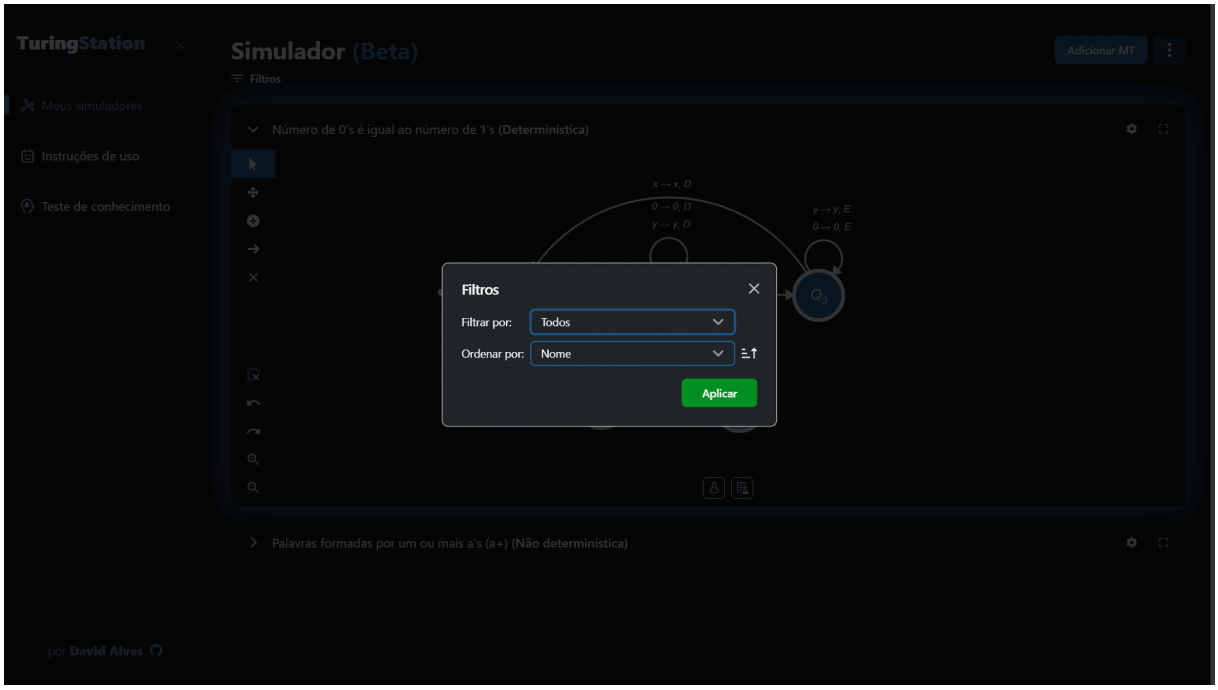


Fonte: Elaborada pelo autor.

Na área principal é posível filtrar simuladores por nome, data de criação, ou data de modificação, para mostrar apenas as variantes de MTs que o usuário deseja no momento, além de poder ordenar as MTs de maneira mais agradável ao usuário. A Figura 19 mostra a caixa de

diálogo onde o usuário pode utilizar os filtros da página Meus Simuladores.

Figura 19 – Modal para utilização de filtros na página Meus simuladores.



Fonte: Elaborada pelo autor.

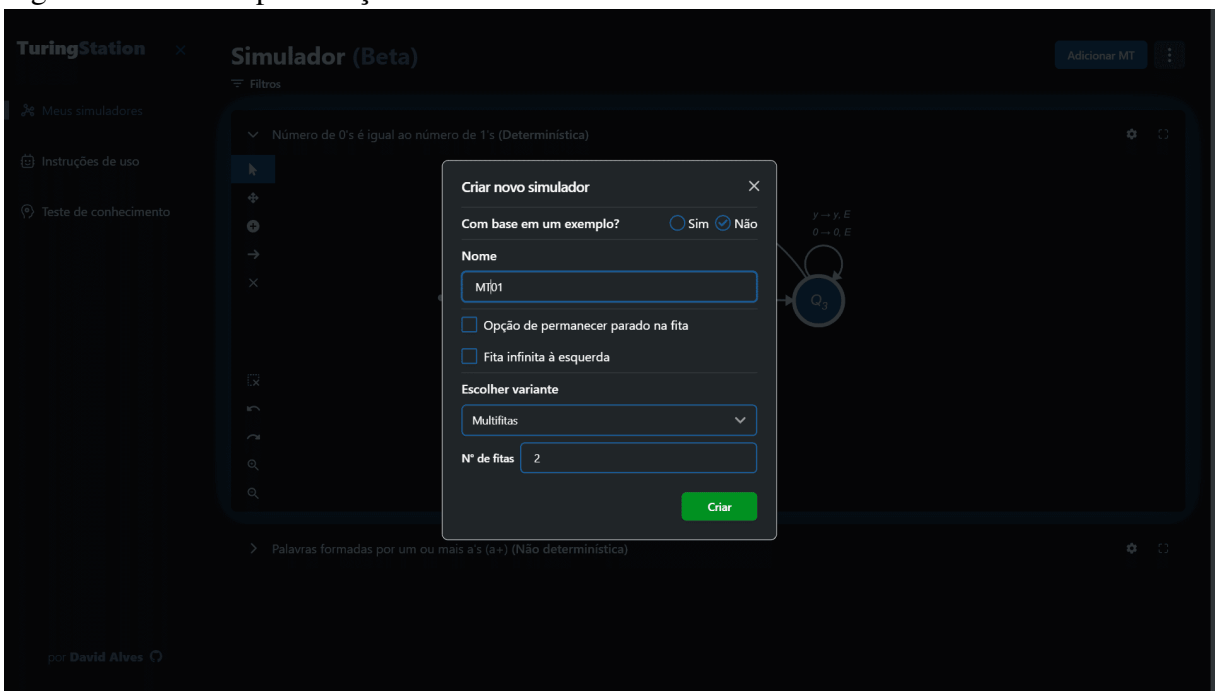
O botão **Adicionar MT** pode ser utilizado para adicionar novos simuladores. Ao clicá-lo, uma caixa de diálogo aparecerá (observe a Figura 20) permitindo ao usuário definir as características da máquina, escolhendo nome e variante, podendo por exemplo combinar até mesmo certas variantes, como MT com opção de parada e MT multifitas. Neste modal também é possível criar uma MT com base em um arquivo importado ou algum exemplo disponibilizado pela própria plataforma (observe a Figura 21).

Ao lado do botão **Adicionar MT**, é possível notar que existe um botão responsável por mostrar um menu denominado **Menu de ações**. Neste menu, o usuário pode habilitar ou desabilitar as dicas que a ferramenta dá, além de poder exportar múltiplas MTs em formato *JavaScript Object Notation* (JSON) para reutilização (importação) futura. Neste menu, o usuário também tem a opção de apagar todos os simuladores em poucos cliques. A Figura 22 mostra o menu de ações da página Meus simuladores.

6.1.1 Simulador

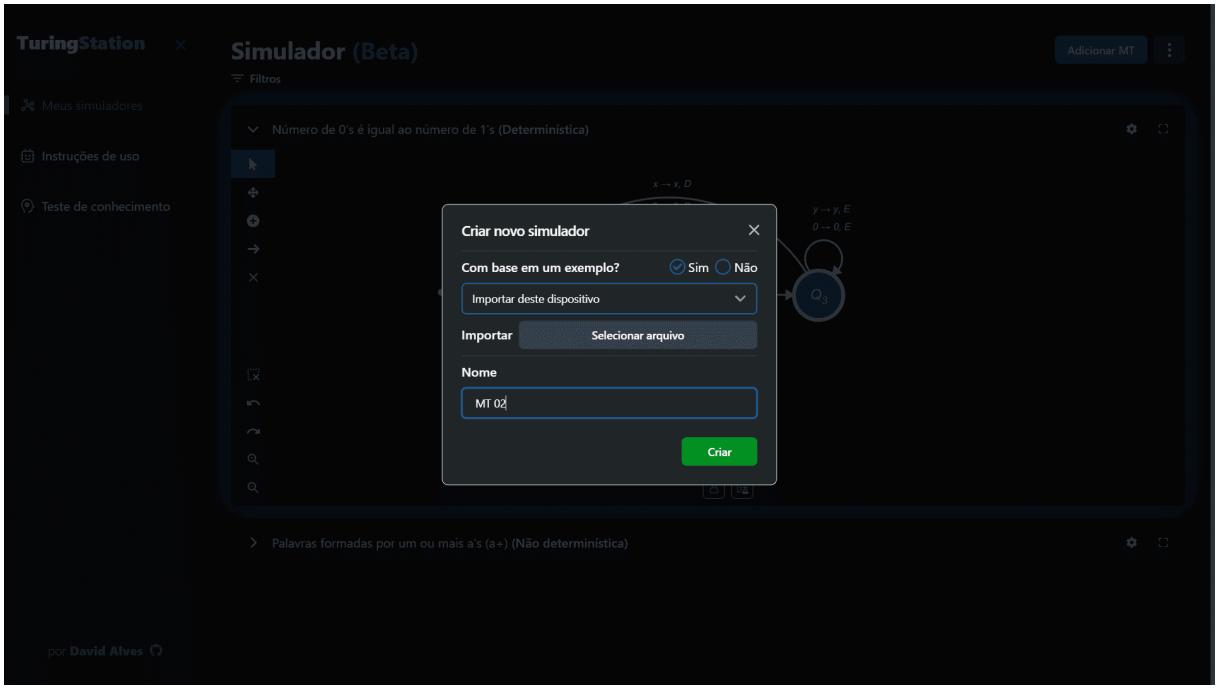
Um simulador na Turing Station é a área onde o usuário pode criar e testar MTs e variantes. Nesta área o usuário pode, por meio de cliques ou toques na tela, criar estados e

Figura 20 – Modal para adição de novos simuladores.



Fonte: Elaborada pelo autor.

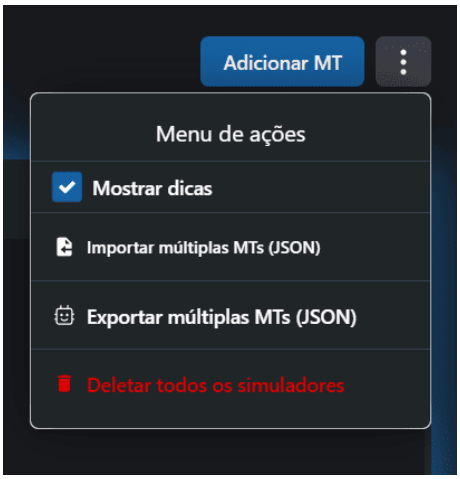
Figura 21 – Adição de simulador baseada em um exemplo.



Fonte: Elaborada pelo autor.

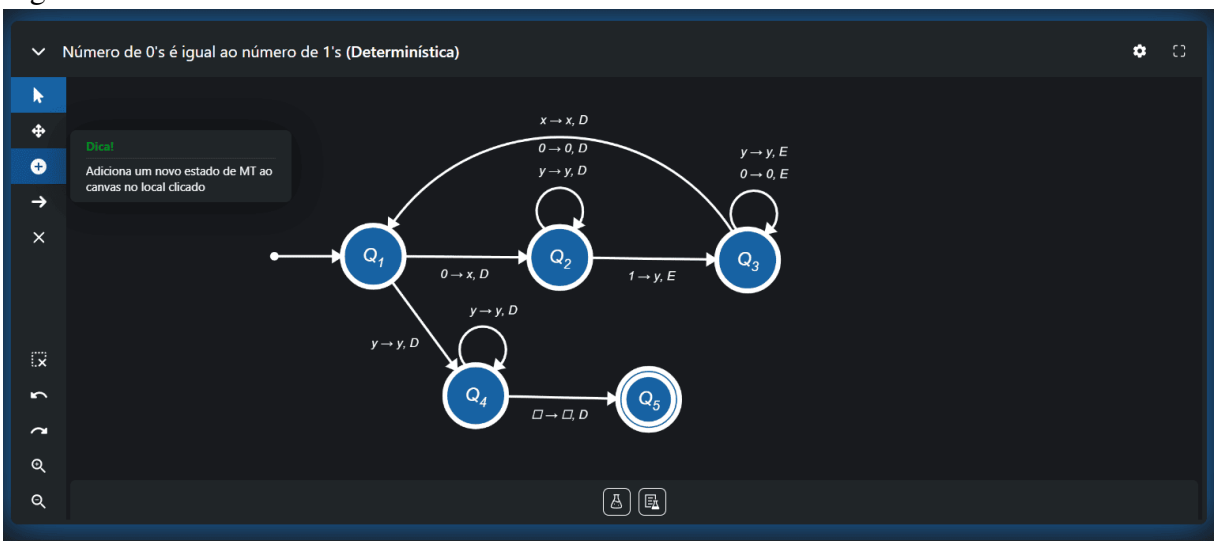
transições de forma interativa e rápida. Estes simuladores foram pensados para serem intuitivos e fáceis de usar. Ao marcar a opção "Mostrar dicas" no menu de ações, o usuário poderá ver dicas do que pode fazer com cada opção dentro do simulador ao passar o ponteiro do *mouse* sobre a mesma. A Figura 23 mostra um simulador juntamente com uma dica flutuante sobre o botão "Adicionar estados".

Figura 22 – Menu de ações da página Meus simuladores.



Fonte: Elaborada pelo autor.

Figura 23 – Simulador de MT com dicas de uso.



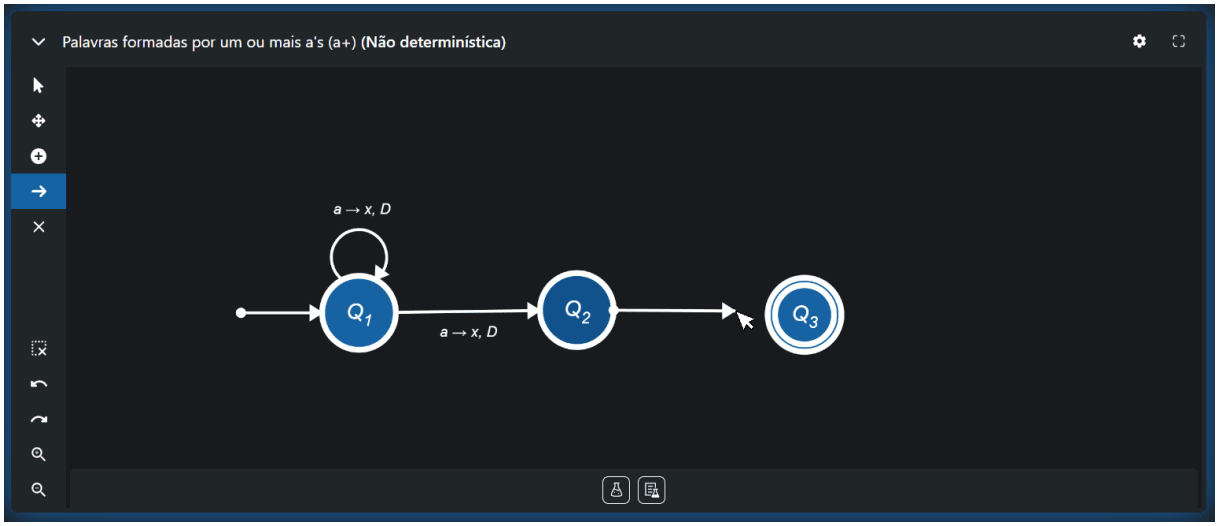
Fonte: Elaborada pelo autor.

Para poder realizar ações dentro da área interativa de criação de MTs, o usuário necessita selecionar a função desejada no menu lateral do simulador. Por exemplo, a primeira opção, indicada por um ícone apontador, pode ser utilizada para seleção e movimentação de elementos dentro da área interativa. As funções dos outros botões envolvem criação, edição e deleção de elementos (estados e transições), limpeza da área interativa, desfazer ou refazer ações, aproximar e afastar (*zoom in/out*). Algumas destas ações possuem atalhos para facilitar a usabilidade, como por exemplo a ação de desfazer, que pode ser realizada ao pressionar a tecla Ctrl + z (*Control* + z) no teclado.

Um exemplo da simplicidade da criação de MTs na Turing Station está na parte de criação de transições. Para isso, o usuário necessita apenas selecionar a opção de criar

transição no menu lateral do simulador, clicar sobre um estado origem, e arrastar o cursor até o estado destino. A Figura 24 demonstra a utilização da funcionalidade de arrasta e solta para a criação de transições no simulador, onde a origem da transição é o estado Q_2 e o destino é o estado Q_3. Ao realizar a ação anterior, uma pequena caixa de diálogo denominada "Caixa de transições" aparecerá. Neste modal deve-se definir a regra de transição escolhendo o símbolo lido, símbolo escrito, e movimento da cabeça de fita. A Figura 25 mostra um exemplo de caixa de transição para uma MT qualquer.

Figura 24 – Criação de transições no simulador.



Fonte: Elaborada pelo autor.

Figura 25 – Caixa de transição no simulador.

Fonte: Elaborada pelo autor.

No topo direito do simulador, juntamente ao botão "Tela cheia", há o menu de configurações do simulador. Neste menu o usuário pode renomear, fazer uma cópia (duplicar) ou até mesmo apagar um simulador. Além disso, o usário tem a opção de exportar a MT em formato de imagem, ou como arquivo JSON, que pode ser posteriormente importado para reutilização. A Figura 26 apresenta o menu de configurações do simulador.

Na Turing Station, o simulador conta com dois tipos de testes (visualização da

Figura 26 – Menu de configurações do simulador.

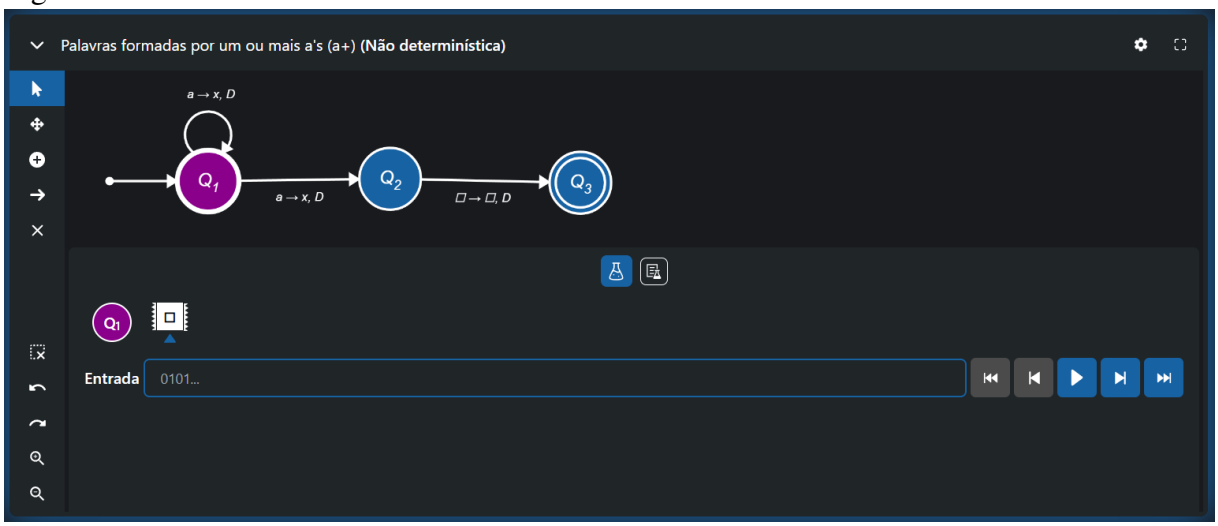
Fonte: Elaborada pelo autor.

computação) para as MTs, sendo uma área para teste unitário e outra para a execução de múltiplos testes de uma única vez. Cada tipo de teste possui seus benefícios e utilidades, sendo o teste unitário ideal para estudo do processo que a computação leva até ser concluída, e o multiteste para o teste rápido de uma MT.

A área de testes unitários pode ser acessada ao clicar no botão localizado na base do simulador, cujo o símbolo é um frasco químico. Nela o usuário pode observar o passo a passo da computação de uma palavra pela MT observando sua fita e/ou observando o diagrama de estados presentes na área interativa, que modifica a cor do estado atual da computação. Além disso, o usuário pode controlar a execução da simulação, podendo avançar passo a passo, pular para o final da execução, retroceder ou simular de forma automática utilizando os botões de controle. A Figura 27 mostra a área de testes unitários no simulador.

A área denominada "Multiteste" pode ser utilizada para o teste de várias palavras ao mesmo tempo. Nela o usuário não consegue ver o passo a passo da computação das palavras testadas, mas recebe um *feedback* imediato com relação as palavras que são aceitas ou rejeitadas pela MT criada. Utilizando os campos de textos e botões de ação, o usuário pode adicionar, remover, e editar as palavras a serem testadas. O resultado do teste é apresentado pelos ícones à esquerda dos campos de texto, que representam a aceitação ou rejeição daquela palavra. A

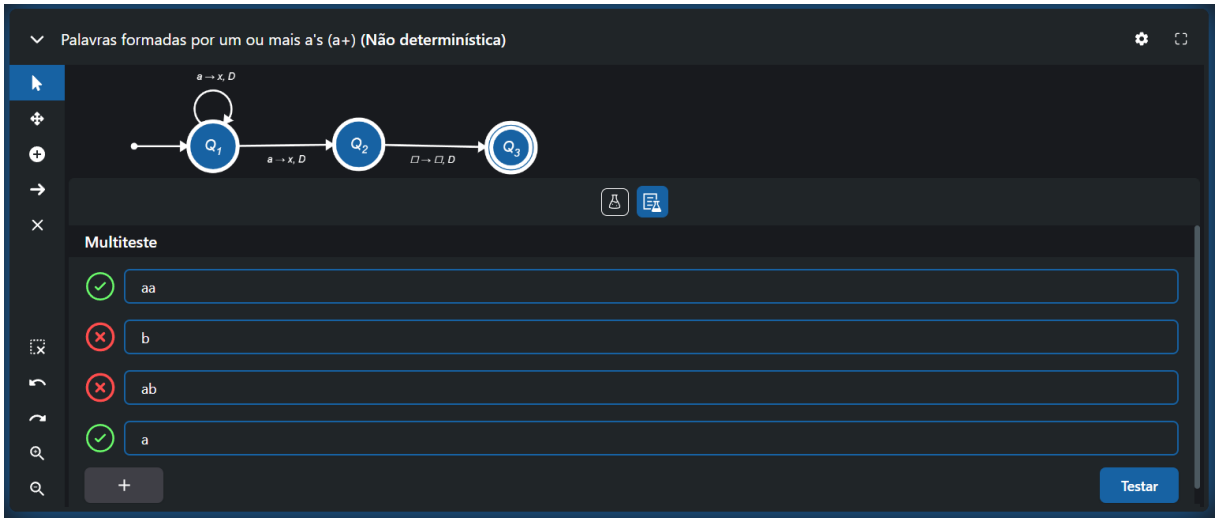
Figura 27 – Área de testes unitários no simulador.



Fonte: Elaborada pelo autor.

Figura 28 mostra o resultado do multiteste das palavras ["aa", "b", "ab", "a"] na MTND ND_{aa} que aceita a linguagem $A = \{aa^*\}$, ou seja, palavras com um ou mais "a"s.

Figura 28 – Multiteste de palavras sobre a linguagem A no simulador.



Fonte: Elaborada pelo autor.

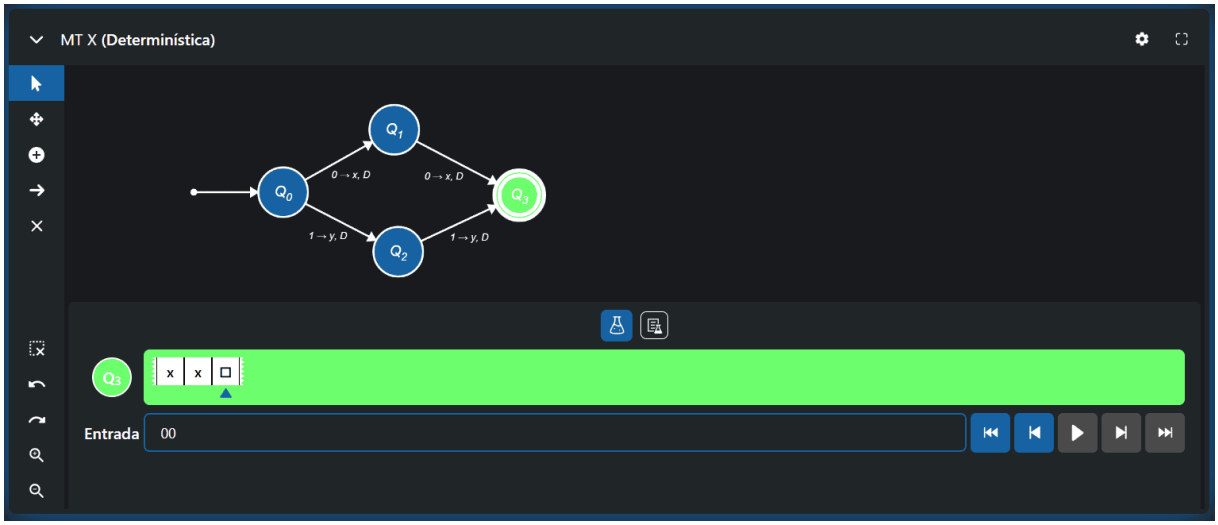
6.1.2 Simulando MTs com a Turing Station

Ao criar uma MT e utilizar a área de testes unitários, o usuário pode visualizar a computação passo a passo de uma palavra qualquer. No caso da MT clássica, como há apenas uma fita, a aplicação mostra a cada iteração a nova configuração desta fita. Ao final da computação, a aplicação mostra de forma clara ao usuário se a palavra computada foi aceita ou não. Quando há a coloração verde no bloco de fita, a palavra foi aceita pela MT testada. Por

outro lado, quando há a colaração vermelha no bloco de fita ao final de uma computação, a palavra foi rejeitada pela MT. Caso após 1000 iterações a computação ainda não tenha terminado, ela é interrompida e o bloco de fita ficará na cor laranja, dando um *feedback* visual ao usuário. Isto é feito para impedir que o navegador web do usuário trave ao realizar computações que possam nunca acabar.

A Figura 29 apresenta o retorno visual da aplicação para o teste da palavra "00" sobre a MT X que aceita a linguagem $K = \{00\} \bigcup \{11\}$, observe que a palavra testada faz parte da linguagem K, sendo aceita. A Figura 30 apresenta o retorno visual da aplicação para o teste da palavra "10" sobre a MT X, observe que a palavra testada não faz parte da linguagem K, sendo rejeitada.

Figura 29 – Exemplo de palavra aceita pela MT X.

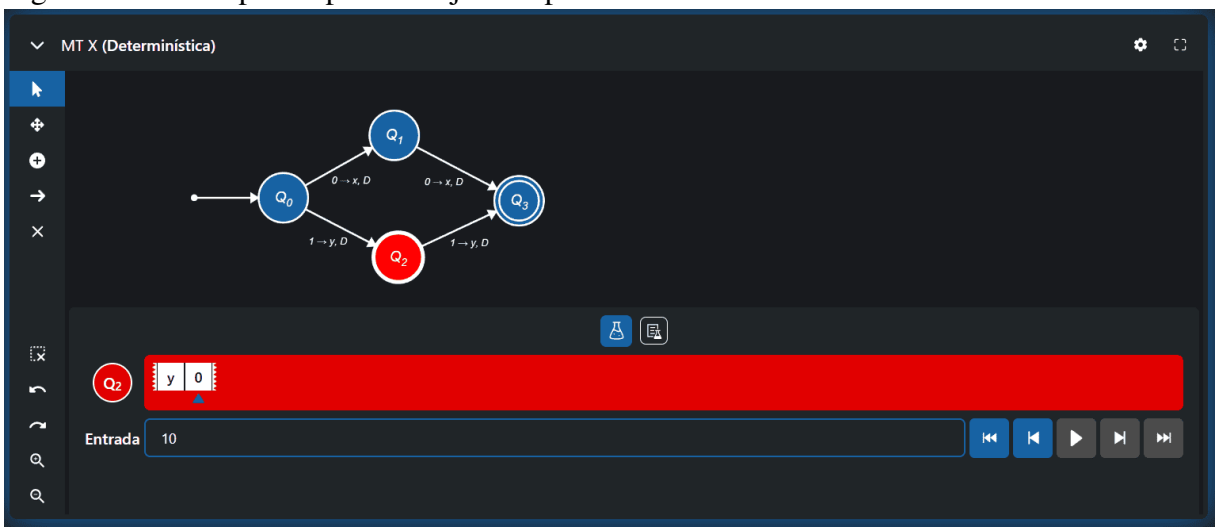


Fonte: Elaborada pelo autor.

6.1.3 Simulando MTNDs com a Turing Station

A simulação de uma MTND na Turing Station funciona de forma semelhante a simulação de uma MT clássica, diferenciando-se apenas por mostrar os ramos de computação a cada iteração. Neste tipo de simulação, os ramos da computação são representados por fitas, sendo uma para cada ramo. Além disso, os possíveis estados em que a computação "se encontra" a cada momento também são coloridos simultaneamente no diagrama de estados. A Figura 31 demonstra o teste da palavra "aaaa" sobre a MTND ND_{aa}, apresentada anteriormente. É possível notar que após o primeiro passo da computação, a MTND ND_{aa} pode estar em dois estados diferentes ao mesmo tempo (estados Q_1 e Q_2, neste exemplo), com a possibilidade

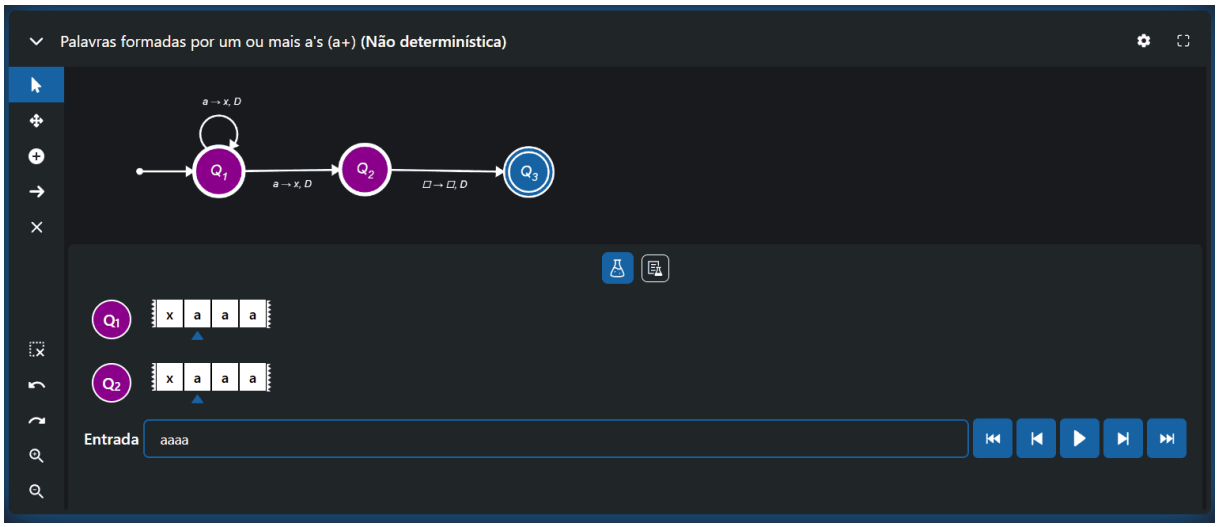
Figura 30 – Exemplo de palavra rejeitada pela MT X.



Fonte: Elaborada pelo autor.

das multiplas cabeças de fita também estarem em lugares distintos, com configurações de fitas distintas.

Figura 31 – Configuração da MTND ND_{aa} após uma computação sobre a palavra "aaaa" no simulador.



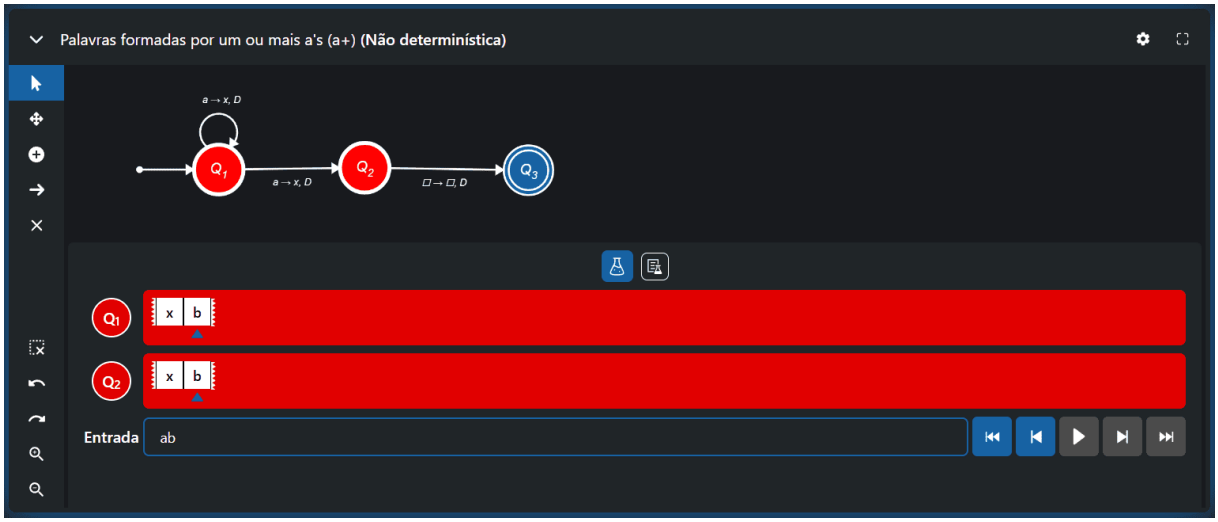
Fonte: Elaborada pelo autor.

Se algum desses ramos entrar em estado de aceitação após um passo, este ficará marcado como verde e a computação será aceita, parando-a. Já se em algum momento, todos os ramos atuais forem rejeitados, todos eles ficarão com uma cor vermelha e representarão a rejeição da palavra computada. A Figura 32 demonstra a configuração das fitas nos dois ramos de computação da MTND ND_{aa}, no momento da sua aceitação sobre a palavra "aaaa". Já a Figura 33 demonstra a configuração das fitas nos ramos de computação da MTND ND_{aa}, no momento da sua rejeição sobre a palavra "ab".

Figura 32 – Exemplo de palavra aceita pela MTND ND_{aa} no simulador.

Fonte: Elaborada pelo autor.

Figura 33 – Exemplo de palavra rejeitada pela MTND ND_{aa} no simulador.



Fonte: Elaborada pelo autor.

6.1.4 Simulando MT multifitas com a Turing Station

A simulação de uma MT mulfitas na Turing Station também funciona de forma semelhante a simulação de uma MT clássica, modificando apenas a sua apresentação. Neste tipo de simulação, a aplicação mostra a configuração das n fitas da MT multifitas em cada momento da computação, sendo n a quantidade de fitas escolhidas pelo usuário.

A rejeição de uma palavra pela MT multifitas é apresentada na aplicação como uma marcação vermelha no bloco de fitas. Já a aceitação de uma palavra é apresentada como uma marcação verde no bloco de fitas. A Figura 34 mostra o teste da palavra "abba" sobre a MT multifitas M_{ab} que aceita palavras pertencentes a linguagem $L_{ab} = \{a^n b^n \mid n >= 0\}$, ou seja, o número de "a"s é igual ao número de "b"s. Observe que a palavra "abba" pertence a linguagem

L_{ab}, sendo aceita. A Figura 35 mostra o teste da palavra "baa" sobre a MT multifitas M_{ab}, observe que a palavra "baa" não pertence a linguagem L_{ab}, sendo rejeitada.

Figura 34 – Exemplo de palavra aceita pela MT multifitas M_{ab} no simulador.

Fonte: Elaborada pelo autor.

Figura 35 – Exemplo de palavra rejeitada pela MT multifitas M_{ab} no simulador.

Fonte: Elaborada pelo autor.

6.1.5 Simulando outras variantes de MT com a Turing Station

Como explicado no Capítulo 2, a Turing Station também permite a criação de MTs com fita finita à esquerda e com opção de parada. A MT com fita finita à esquerda não altera nenhum aspecto visível da aplicação, impactando somente na dinâmica da computação da MT. Já a MT com opção de parada adiciona um novo botão na caixa de criação de transições, o botão "P", que permite ao usuário definir esta funcionalidade. A Figura 36 demonstra a presença do

botão "P" na caixa de criação de transições de uma MT com opção de parada no simulador.

Figura 36 – Caixa de transições com botão de parada.

Fonte: Elaborada pelo autor.

6.2 Página Instruções de Uso

A página **Instruções de uso** é a tela onde o usuário pode aprender a utilizar todos as funcionalidades da plataforma Turing Station por meio de vídeos e listas de atalhos. A Figura 37 apresenta a página **Instruções de uso**.

Figura 37 – Página Instruções de uso.

Fonte: Elaborada pelo autor.

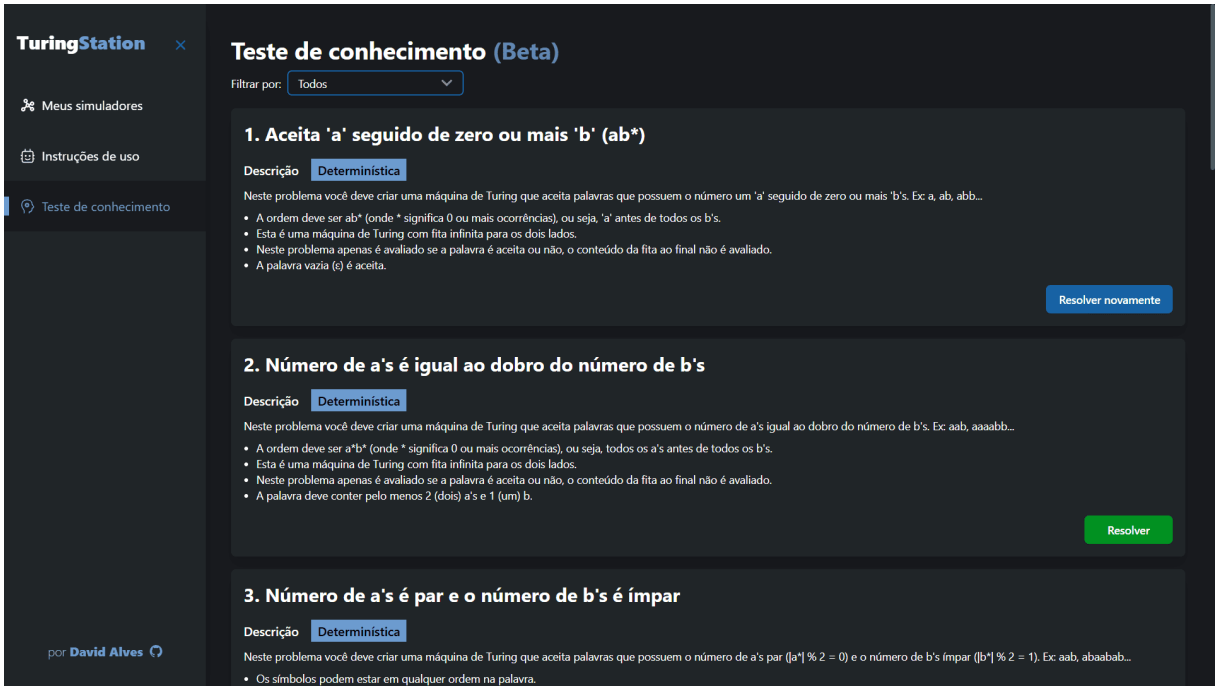
Nesta tela são apresentados vídeos demonstrando o uso da aplicação tanto em dispositivos de mesa, como em dispositivos móveis. Também são listados todos os atalhos de teclados que ajudam a otimizar a criação e edição das MTs, como por exemplo o atalho para

desfazer uma ação *ctrl* + *z*. Além disso, também há a listagem de atalhos de texto, que servem para a adição de letras gregas nas regras de transição das MTs, como por exemplo o atalho *\alpha* que adiciona a letra grega α.

6.3 Página Teste de Conhecimento

A página **Teste de conhecimento** é onde o usuário tem a oportunidade de praticar a resolução de questões relacionadas a MTs. Nesta tela o usuário visualiza uma lista de questões, onde cada uma é apresentada com um título, variante da MT, uma descrição completa e um botão para direcioná-lo a área de resolução da questão específica escolhida. A Figura 38 apresenta a página **Teste de conhecimento**.

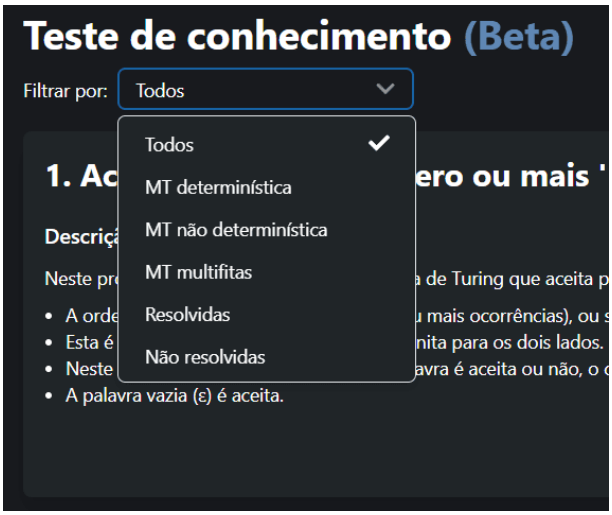
Figura 38 – Página Teste de conhecimento.



Fonte: Elaborada pelo autor.

Para encontrar um tipo de questão específica, o usuário pode filtrar a lista de questões utilizando o filtro no topo da página. Neste filtro, o usuário pode escolher visualizar apenas questões sobre MTs determinísticas, MTs não determinísticas, MTs multifitas, além de filtrar por questões resolvidas e ainda não resolvidas. A Figura 39 mostra o filtro disponibilizado na página **Teste de conhecimento**.

Figura 39 – Filtro de questões na página Teste de conhecimento.



Fonte: Elaborada pelo autor.

6.3.1 Resolvendo um teste de conhecimento

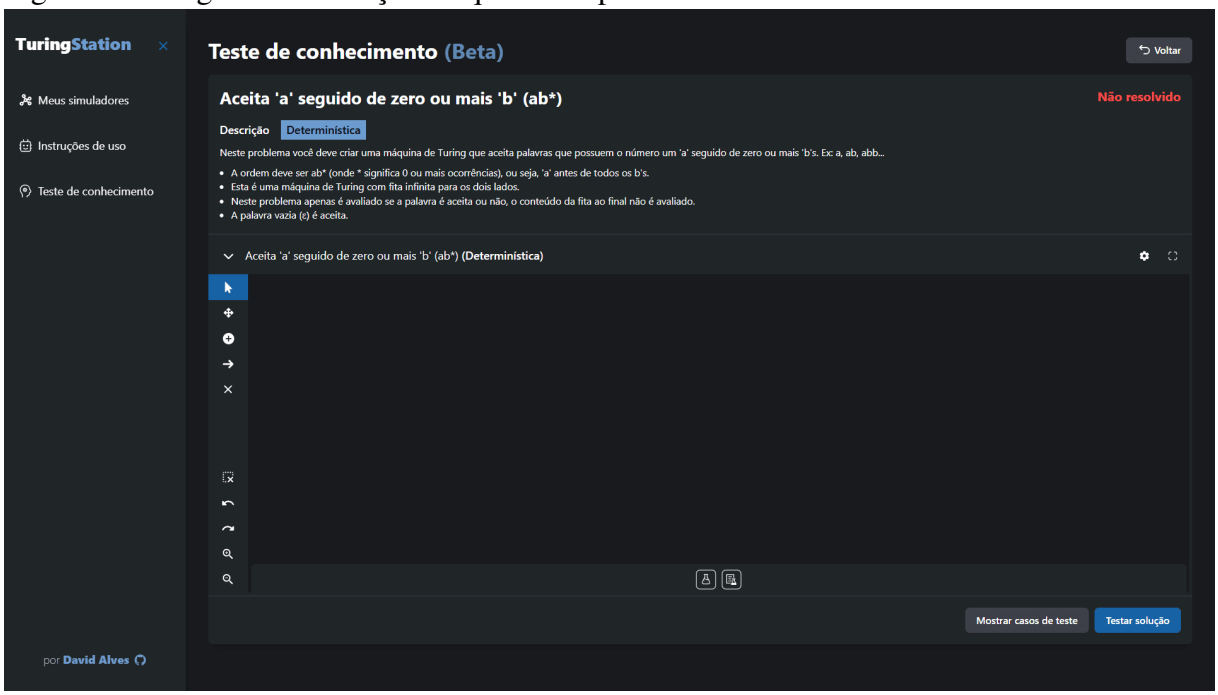
Ao selecionar uma questão, o usuário é redirecionado a uma página específica para a resolução desta questão. Assim como na página de listagem de questões, o usuário pode visualizar o título, a variante da MT que será utilizada, a descrição da questão, o status de resolução, e um simulador disponibilizado especialmente para a resolução desta questão. A Figura 40 mostra um exemplo de página de resolução de questão na plataforma Turing Station, com a questão de título *"1. Aceita 'a' seguido de zero ou mais 'b' (ab*)"* selecionada.

O simulador apresentado nesta área funciona exatamente como o simulador apresentado na página **Meus simuladores**, com a pequena diferença de não permitir a importação de uma MT, e a sua renomeação, duplicação ou exclusão.

Logo abaixo do simulador, há uma área onde são listados os casos de teste da questão escolhida. Os casos de testes são palavra específicas que foram disponibilizadas para confirmar que a máquina desenvolvida pelo usuário realmente resolve aquela questão. A coluna **Palavra** lista as palavras que serão testadas, a coluna **Resultados** lista os resultados obtidos ao clicar em **Testar solução**, já a coluna **Resultados esperados** lista os resultados que são esperados ao computar uma palavra em específico. A Figura 41 mostra a área de listagem de casos de teste, juntamente com os botões "Esconder casos de teste", utilizados para ocultá-los, e o botão "Testar solução", utilizado para realizar o teste dos casos.

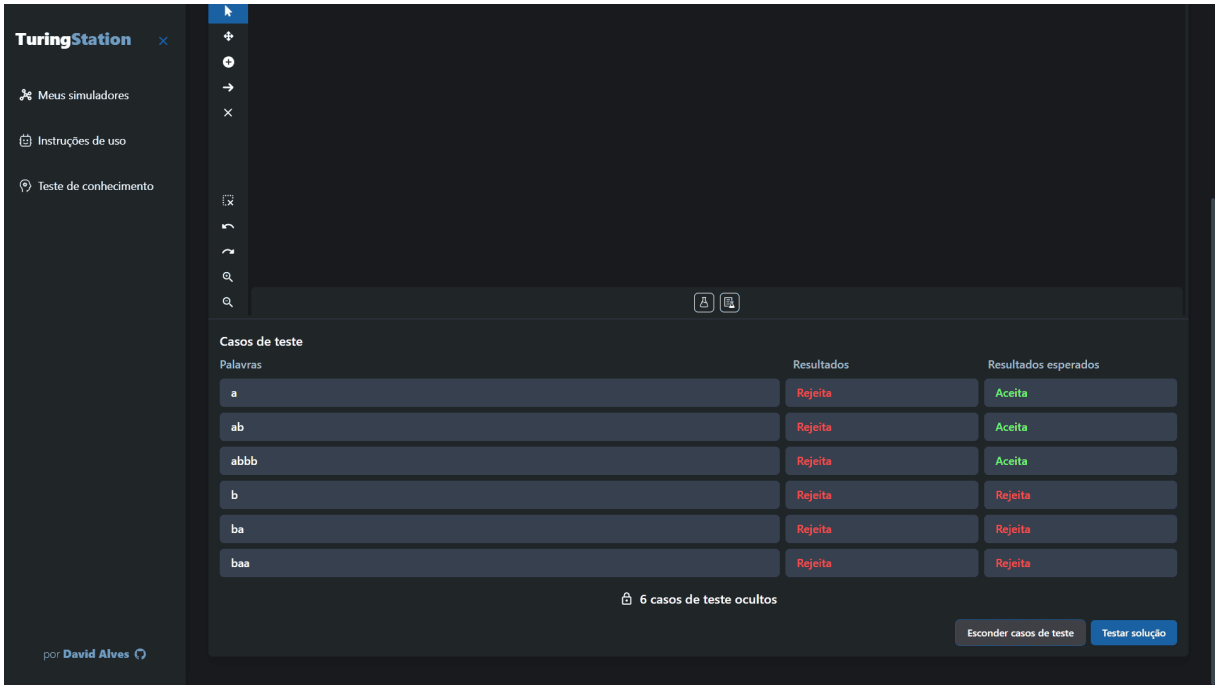
Ainda na Figura 41, é possível ver que há também casos de testes ocultos. Estes tipos de casos de teste funcionam exatamente como os testes visíveis. Eles existem para impedir que

Figura 40 – Página de resolução de questão específica.



Fonte: Elaborada pelo autor.

Figura 41 – Área de listagem de casos de teste para uma questão específica.



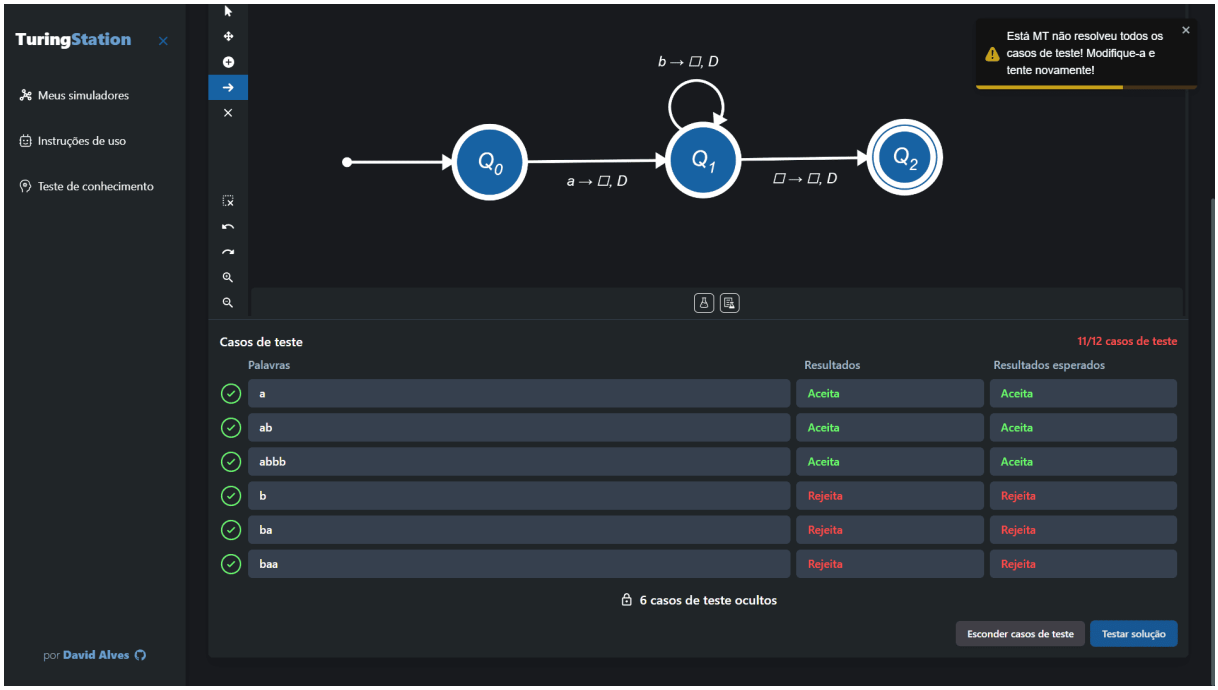
Fonte: Elaborada pelo autor.

o usuário crie uma MT que resolva apenas casos isolados mas que não reconhece a linguagem completa.

Ao clicar em "Testar solução", o usuário terá uma resposta visual juntamente com uma mensagem rápida o informando se a sua solução foi válida ou não. Caso a sua solução sejá válida, o status da questão passará de **Não resolvida** para **Resolvida**. A Figura 42 mostra um

exemplo de feedback em uma tentativa de resolução da questão *"1. Aceita 'a' seguido de zero ou mais 'b' (ab*)"*, onde 11 casos de teste, de um total de 12, passaram no teste . Isto acontece porque a questão exigia que a palavra vazia (ε) fosse aceita, enquanto a MT desenvolvida não engloba este caso.

Figura 42 – *Feedback* para tentativa de resolução de uma questão específica.



Fonte: Elaborada pelo autor.

6.4 Responsividade na Turing Station

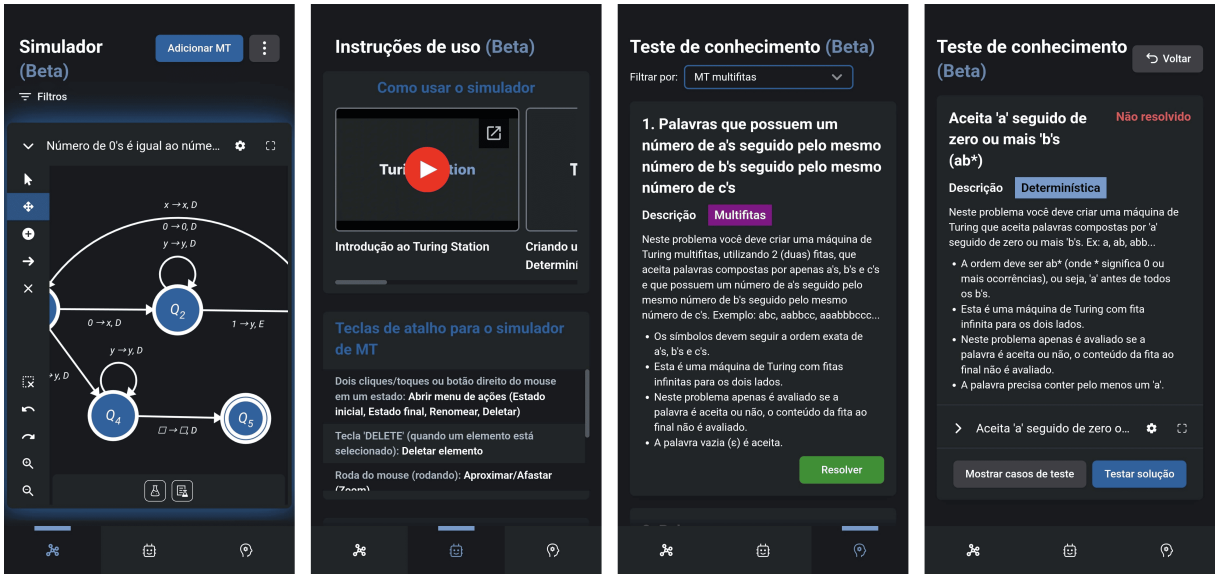
Segundo Ward (2017), no contexto web, o *design* responsivo é capacidade de adaptar as páginas a qualquer dispositivo, respeitando tamanhos de telas e resoluções. Neste contexto, um dos principais objetivos da Turing Station foi possibilitar a sua utilização em diversos dispositivos diferentes, como *smartphones, tablets* e computadores de mesa. Essa capacidade de utilização em dispositivos diversos pode ampliar a quantidade de usuário que utilizarão a aplicação, visto que em sala de aula muitas vezes é mais difícil a utilização de dispositivos maiores como *laptops* ou computadores de mesa.

Para a idealização do *layout* da aplicação para diferentes dispositivos, utilizou-se de análise da estrutura das telas das ferramentas similares. Embora a maioria, neste caso todas as analisadas, não disponham de responsividade, foi possível combinar aspectos do JFLAP, que pode ser utilizado em dispositivos de mesa, e do FLApp, que pode ser utilizado por dispositivos

móveis com sistema operacional *Android*.

A versão para dispositivos móveis da Turing Station apresenta exatamente as mesmas funcionalidades presentes na versão para computadores de mesa, contudo, houve a necessidade de diversas adaptações de *layout* para que fosse possível apresentar uma ótima experiência para o usuário da versão móvel. A Figura 43 apresenta as quatro principais telas do *website* na versão para dispositivos móveis. Nesta versão, o menu de navegação lateral foi posicionado na parte inferior da tela, para que fosse possível apresentar uma versão similar a versão nativa de um aplicativo móvel. Algumas características de *layout* também precisaram ser adaptadas, mas sem prejudicar as funcionalidades propostas.

Figura 43 – Principais telas do *website* na versão *mobile*.



Fonte: Elaborada pelo autor.

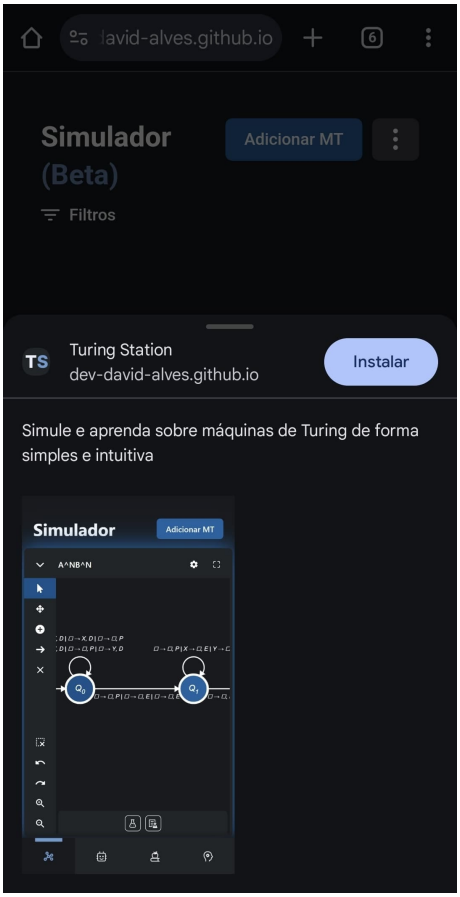
6.5 Aplicação Web progressiva

Outra característica que diferencia a Turing Station das demais ferramentas similares citadas é o advento da denominada arquitetura "web progressiva". Aplicativos Web Progressivos (AWP) oferecem, além da experiência conhecida de se utilizar um *website*, a experiência de um aplicativo instalado de forma nativa em uma máquina, como os aplicativos encontrados na loja de aplicativos *Android*, a *Play Store*.

Por poderem ser instalados localmente, os AWP podem ser utilizados completamente de forma *off-line* (sem conexão com a *internet*), além de possuir tempos de carregamentos quase que instantâneos. O AWP utiliza um arquivo denominado *Manifest* que guarda informações

sobre ícones, orientação de dispositivo, imagens de fundo, e mais, para tornar possível o salvamento na memória do navegador. Jutamente ao *Manifest*, o AWP também utiliza um arquivo denominado *Service Worker*, que permite a realização de requisições de redes, habilitando inclusive a possibilidade de instalação do *website* localmente (Biørn-Hansen *et al.*, 2017; Hume, 2017). A Figura 44 demonstra um exemplo de mensagem de chamada para instalação da Turing Station em um dispositivo *Android* por meio do navegador *Google Chrome*.

Figura 44 – Instalando a Turing Station localmente em um *smartphone Android*.



Fonte: Elaborada pelo autor.

Ao instalar um AWP, um ícone de aplicativo aparecerá na área de trabalho do dispositivo. A aparência do *website* se torna idêntica a de um aplicativo nativo, apresentando tela de carregamento e ocultando componentes de tela que não são importantes para a utilização direta fora de um navegador web (Biørn-Hansen *et al.*, 2017; Hume, 2017).

7 AVALIAÇÃO DA FERRAMENTA COM USUÁRIOS

Após o seu desenvolvimento, a aplicação Turing Station foi avaliada em sala de aula por alunos da disciplina de Teoria da Computação na Universidade Federal do Ceará - Campus Quixadá. Para isto, o professor da disciplina disponibilizou uma lista de questões relacionadas a construção de MTs e variantes para que os alunos resolvessem, dando-os a opção de utilizar ou não a aplicação.

Passado-se o período de aproximadamente 3 semanas destinado para a resolução da lista de exercícios por parte dos alunos, foi disponibilizado um formulário *on-line* por meio da ferramenta *Google Forms*[1] para obteção de *feedbacks* sobre a experiência dos discentes ao utilizar a aplicação. O preechimento deste formulário pelos alunos foi totalmente opcional, obtendo assim um total de 11 respostas.

Para uma melhor organização, este formulário foi estruturado em 5 seções, sendo a primeira seção relacionada a termos e consentimentos, a segunda relacionada a questões demográficas, a terceira sobre a experiência de uso na aplicação utilizando a Escala de Usabilidade do Sistema (SUS), a quarta relacionada a perguntas mais abertas sobre UX e UI, e a quinta e última foi focada em questões sobre o impacto da Turing Station no aprendizado dos estudantes. As próximas seções deste capítulo detelharão cada uma das seções do formulário, com exceção da primeira seção do formulário visto que esta não se relaciona exatamente com o objetivo de pesquisa deste trabalho. O Apêndice A apresenta todas as questões do formulário divididas por suas respectivas seções.

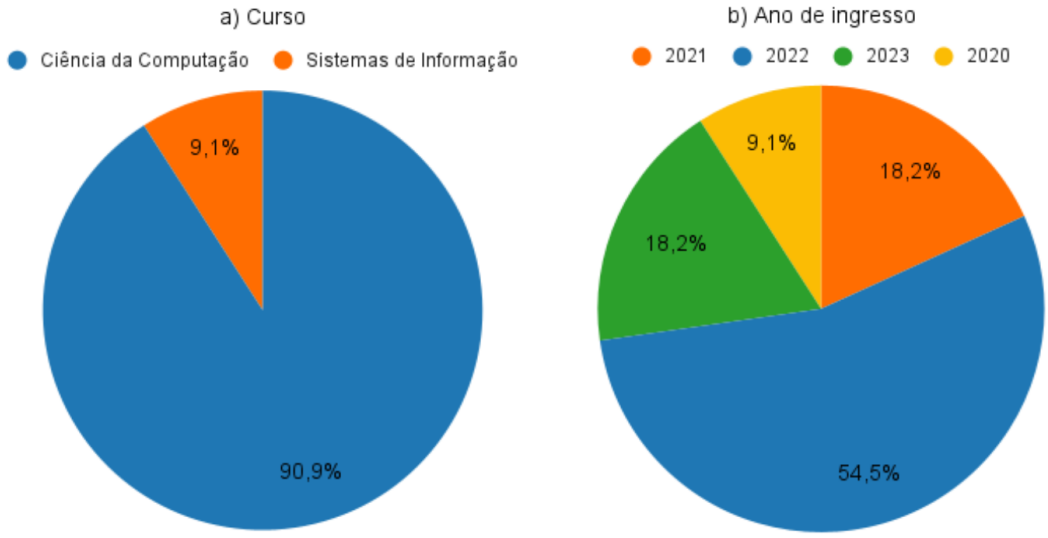
7.1 Questões demográficas

Nesta seção do formulário foram realizadas 2 perguntas para mapeamento do público que utilizou a ferramenta. A primeira pergunta foi sobre o curso que o discente está cursando na universidade. Já a segunda pergunta foi sobre o seu ano de ingresso naquele curso. A Figura 45 mostra, em ordem, os gráficos das duas questões descritas.

Como pode-se observar pelo gráfico (a) da Figura 45, a grande maioria dos discentes, cerca de 91%, fazem parte do curso Ciência da Computação. Outros 9% cursam Sistemas da Informação. Estes números se devem principalmente ao fato da disciplina de Teoria da Computação ser obrigatória para os alunos de Ciência da Computação e opcional (optativa) para

[1] https://docs.google.com/forms/u/0/

Figura 45 – Gráficos que apresentam as respostas para as questões demográficas.



Fonte: Elaborada pelo autor.

alunos de Sistemas da Informação.

Já com o gráfico (b) da Figura 45, percebe-se que todos os participantes ingressaram em seus cursos a partir do ano 2020, sendo que a grande maioria ingressou especificamente no ano 2022. Este dado demonstra que os usuários, principalmente os que ingressaram a mais tempo, possivelmente possuem experiência com plataformas web em geral. Além disso, os discentes pertecentes ao curso de Ciência da Computação possuem uma disciplina denominada Linguagens Formais e Autômatos, sendo esta obrigatória e realizada anteriormente à disciplina de Teoria da Computação, o que dá aos usuários um prévio conhecimento sobre a criação de máquinas abstratas.

7.2 Questões sobre experiência de uso utilizando a Escala de Usabilidade do Sistema (SUS)

A Escala de Usabilidade do Sistema (SUS, do inglês *System Usability Scale*) é um tipo de escala criada para avaliar a usabilidade de um sistema de forma quantitativa. Ela foi desenvolvida por John Brooke (Brooke, 1996) e pode ser utilizada para avaliar os mais diversos tipos de interfaces, como *softwares*, *websites*, *hardwares*, serviços, e outros (Lewis, 2018).

O SUS visa cobrir 3 aspectos principais de usabilidade de um sistema, aplicação ou produto, sendo eles:

– Eficácia: a habilidade dos usuários poderem completar suas tarefas e atingir seus objetivos ao utilizar o sistema e a qualidade dos resultado retornados;

- Eficiência: a quantidade de recursos consumidos ao realizar as tarefas;
- Satisfação: a reação subjetiva dos usuários ao utilizar o sistema.

A mensuração da eficácia e da eficiência dos sistemas pode variar dependendo dos tipos de tarefas realizadas e até mesmo da forma de medição. Já a satisfação, depende totalmente da experiência de uso e da interpretação subjetiva da usabilidade pelo o usuário (Brooke, 1996; Lewis, 2018).

O SUS é um questionário composto por 10 itens, onde cada um deve ser respondido por meio da escala de Likert (Likert, 1932) que varia de 1 a 5, onde 1 é **Discordo totalmente** e 5 é **Concordo totalmente**. A seguir, estão listados os 10 itens presentes no SUS:

1. Eu acho que gostaria de usar esse sistema com frequência;
2. Eu acho o sistema desnecessariamente complexo;
3. Eu achei o sistema fácil de usar;
4. Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema;
5. Eu acho que as várias funções do sistema estão muito bem integradas;
6. Eu acho que o sistema apresenta muita inconsistência;
7. Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente;
8. Eu achei o sistema atrapalhado de usar;
9. Eu me senti confiante ao usar o sistema;
10. Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.

A Figura 46 apresenta os resultados obtidos nos itens relacionados ao SUS para o questionário aplicado em sala de aula. Cada gráfico na figura representa uma questão do SUS em formato de gráfico de setores, apresentando as quantidades de respostas para o respectivo item na escala de Likert.

Utilizando o cálculo adaptado de (Lewis, 2018) para calcular o resultado geral do SUS, deve-se considerar de forma individual cada resposta de usuário e realizar o calculo da média dessas respostas. Primeiro, define-se que a posição dos itens na escala de Likert representarão pontuações que serão utilizadas no cálculo, onde Discordo totalmente vale 0, Discordo parcialmente vale 1, e assim até Concordo totalmente que valerá 4. Para as questões ímpares (1,3,5,7 e 9), a contribuição da afirmação dar-se-á pelo valor de sua pontuação. Para as questões pares (2,4,6,8 e 10), a contribuição da questão dar-se-á por 4 menos a pontuação da questão. Ao final, soma-se todas as contruibuições das pontuações calculadas e multiplica-se o

Figura 46 – Resultados obtidos nas questões do modelo SUS.

Fonte: Elaborada pelo autor.

resultado por $2,5$. A formula final do SUS individual é $SUS = 2,5 * (S_1 + (4 - S_2) + S_3 + (4 - S_4) + S_5 + (4 - S_6) + S_7 + (4 - S_8) + S_9 + (4 - S_{10}))$, onde S_i representa a pontuação da questão de número i.

Calculando o SUS geral para o questionário aplicado em sala de aula, obteve-se $SUS = 87,73$. Este resultado pode ser mapeado utilizando a tabela apresentada por Sauro e Lewis (2016, p. 204), que mapeia os intervalos de valores resultantes do SUS para letras, sendo $A+$ a melhor nota possível, e F a pior nota possível. A Figura 47 apresenta a tabela utilizada para mapeamento do resultado do SUS.

Figura 47 – Mapa para resultados do SUS.

Faixa de pontuação SUS	Nota	Faixa percentil
84,1 - 100	A+	96 - 100
80,8 - 84	A	90 - 95
78,9 - 80,7	A-	85 - 89
77,2 - 78,8	B+	80 - 84
74,1 - 77,1	B	70 - 79
72,6 - 74	B-	65 - 69
71,1 - 72,5	C+	60 - 64
65 - 71	C	41 - 59
62,7 - 64,9	C-	35 - 40
51,7 - 62,6	D	15 - 34
0 - 51,7	F	0 - 14

Fonte: Adaptado de Sauro e Lewis (2016, p. 204).

Baseado na Figura 47, a aplicação Turing Station obteve a nota $A+$ na potuação geral do SUS. Este resultado demonstrou que a aplicação atendeu de forma excelente a maior parte dos requistos de usabilidade esperados, destacando-se principalmente na simplicidade de uso, eficiência das interações e satisfação geral dos usuários. Esse desempenho reflete um alto grau de alinhamento com os padrões de *design* centrados no usuário, garantindo uma experiência intuitiva e funcional.

7.3 Questões sobre Experiência de Usuário (UX) e Interface de Usuário (UI)

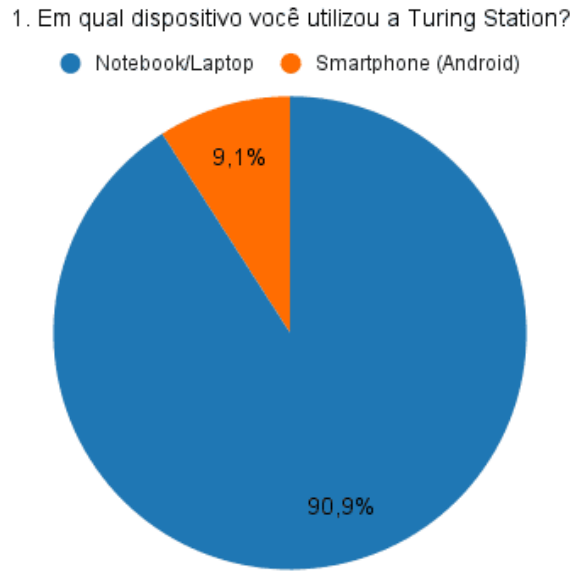
Nesta seção foram apresentadas 7 questões com foco em descobrir o nível de satisfação geral com a interface e a experiência de usuário da aplicação Turing Station. A primeira questão era de múltipla escolha, as afirmações 2 a 5, assim como no questionário SUS, deviam ser respondidas por meio da escala de Likert, e as 2 últimas questões são de texto aberto, além de serem opcionais. A seguir, estão listadas as 7 questões presentes nesta seção:

1. Em qual dispositivo você utilizou a Turing Station?
2. Eu acho que o design da interface da Turing Station contribuiu para uma experiência mais agradável;
3. Eu acho que as instruções fornecidas pela ferramenta são claras e suficientes para iniciar e realizar simulações de Máquinas de Turing;
4. Eu acho que navegar entre as funcionalidades da Turing Station foi intuitivo;
5. Eu acho que os tempos de resposta e a estabilidade da Turing Station atenderam às expectativas;

6. O que você mais gostou na Turing Station?
7. Há algo que poderia ser melhorado para tornar a Turing Station mais útil ou intuitiva? Se sim, o que seria?

A Figura 48 mostra os resultados obtidos para a pergunta *"1. Em qual dispositivo você utilizou a Turing Station?"*. Estes resultados demonstraram que, no momento, os principais dispositivos utilizados foram *Notebooks/Laptops*, com 90,9% de utilização, e *smartphones* com sistema operacional *Android*, com 9,1% de uso. Estes resultados podem ser utilizados para direcionar o foco do desenvolvimento e aprimoramento da interface para estes principais tipos de dispositivos, sem excluir os demais.

Figura 48 – Resultados obtidos para a questão 1.



Fonte: Elaborada pelo autor.

O resultado obtido para a afirmação *"2. Eu acho que o design da interface da Turing Station contribuiu para uma experiência mais agradável"* demonstrou que a interface visual da Turing Station agradou a todos os participantes da pesquisa. Isto aponta que as escolhas de cores, formas, e organização de *layout* foram acertivas, contribuindo para uma melhor experiência com a aplicação. A Figura 49 apresenta os resultados da afirmação 2.

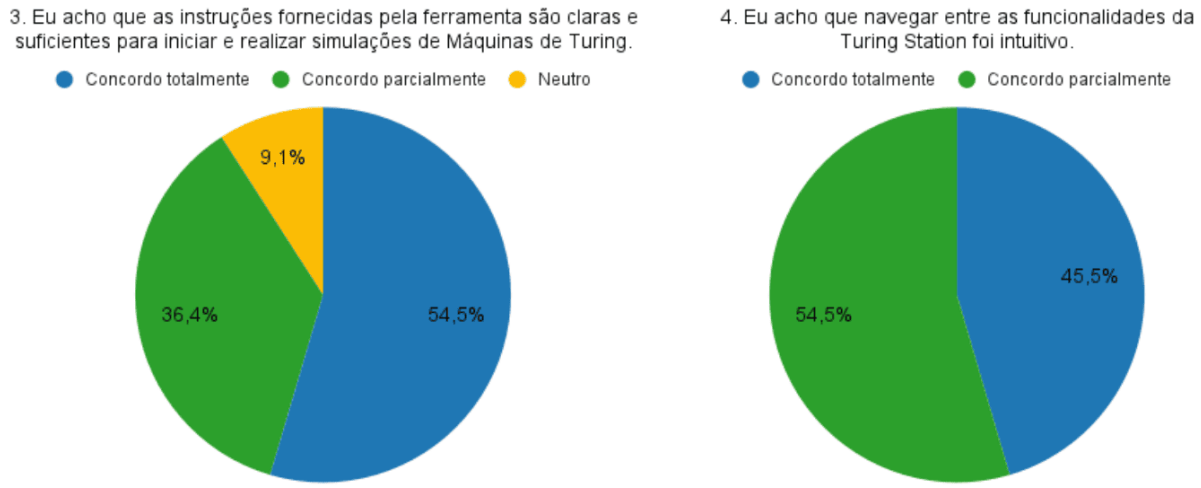
As afirmações 3 e 4 também obtiveram resultados considerados positivos. As respostas para a afirmação *"3. Eu acho que as instruções fornecidas pela ferramenta são claras e suficientes para iniciar e realizar simulações de Máquinas de Turing"* oscilaram entre respostas neutras a totalmente concordantes. Este resultado, embora mostre que a ferrenta possui descrições que contribuem para a sua utilização, também mostrou que há possíveis melhorias a se

Figura 49 – Resultados obtidos para a afirmação 2.

Fonte: Elaborada pelo autor.

fazer, como por exemplo, adicionar mais dicas dispostas pela a ferramenta. Já as respostas para a afirmação *"4. Eu acho que navegar entre as funcionalidades da Turing Station foi intuitivo"* demonstraram que a utilização da Turing Station foi considerada bastante intuitiva, o que aponta que sua estrutura de páginas e organização estão bem apresentadas e distribuídas. Os resultados obtidos pelas afirmações 3 e 4 podem ser observados na Figura 50.

Figura 50 – Resultados obtidos para as afirmações 3 e 4.

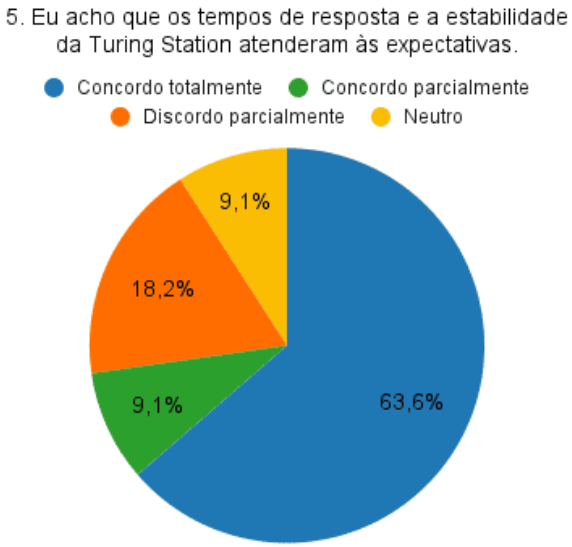


Fonte: Elaborada pelo autor.

Já a afirmação *"5. Eu acho que os tempos de resposta e a estabilidade da Turing Station atenderam às expectativas"* obteve resultados variados, com respostas partindo de Discordo parcialmente até Concordo totalmente. Embora a maior parte das respostas tenham sido

positivas para a afirmação, estes resultados demonstram que a aplicação ainda tem otimizações a serem feitas para melhorar o seu tempo de resposta e a sua estabilidade. Estes aspectos podem impactar negativamente a experiência do usuário e por isso precisam de um foco especial, especialmente no caso de uma aplicação com foco educacional. A Figura 51 apresenta os resultados obtidos para a afirmação 5.

Figura 51 – Resultados obtidos para a afirmação 5.



Fonte: Elaborada pelo autor.

O item *"6. O que você mais gostou na Turing Station?"* foi configurado para receber uma resposta aberta, oferecendo ao usuário a liberdade de expressar sua resposta de maneira detalhada e personalizada. Nele os usuários poderiam descrever os pontos de maior agrado em relação a aplicação, totalmente de forma subjetiva.

As respostas do item 6 mostram que a Turing Station conquistou os usuários pela simplicidade e pela facilidade de uso. Os usuários destacaram como é prático criar e modificar máquinas de Turing, conectar estados e acompanhar os testes com simulações passo a passo bem detalhadas. A interface foi muito elogiada por ser intuitiva, bonita e agradável, além de funcionar bem em dispositivos móveis, o que ajuda em momentos em que o notebook não está disponível. Funcionalidades como a possibilidade de mover o canvas, organizar máquinas mais complexas e realizar multitestes foram vistos como grandes diferenciais. Além disso, também foi mencionado a integração com o conteúdo das aulas e a praticidade da plataforma como pontos que realmente fizeram a diferença. O Apêndice B apresenta cada uma das respostas adicionadas pelos respondentes para o item 6.

O item *"7. Há algo que poderia ser melhorado para tornar a Turing Station mais útil ou intuitivo? Se sim, o que seria?"* também foi configurado para permitir aos usuários descreverem suas opniões de forma aberta e subjetiva. Neste item, os usuários podiam dar opniões para tornar a Turing Station mais útil e intutitiva, embora a maioria dos usuários tenham utilizado este campo para relatar erros ou problemas específicos na aplicação.

As respostas do item 7 apresentam as surgestões dos usuários sobre melhorias para tornar a Turing Station ainda mais prática. Destacaram a necessidade de funcionalidades como a seleção múltipla de estados/objetos e o acréscimo de mais atalhos de teclado, além de um histórico de edição mais completo. Ajustes na funcionalidade de zoom para afetar apenas a área de *design* de MTs, melhorias no desempenho para evitar lentidão, e correções de erros, como na exportação de imagens e na importação de máquinas, também foram mencionados. Por fim, recursos como editar tipos de variantes após a criação e poder copiar múltiplos estados da MT com suas conexões seriam bem-vindos para facilitar o uso. O Apêndice C apresenta cada uma das respostas adicionadas pelos respondentes para o item 7.

Em suma, as respostas para esta seção do formulário permitiram concluir que a aplicação Turing Station atende a grande maioria das expectativas de uso e de interface dos usuários. Todavia, algumas melhorias futuras puderam ser percebidas por meio destas respostas, como a melhoria no tempo de resposta da aplicação, e uma descrição mais detalhada de suas funcionalidades por meio do acréscimo de mais dicas em seus componentes, além do acréscimo de mais atalhos para tornar a criação de MTs mais rápida.

7.4 Questões sobre os impactos da Turing Station no aprendizado

Nesta seção foram apresentados 5 afirmações relacionadas a interpretação subjetiva do usuário sobre os impactos que a aplicação Turing Station teve em seu aprendizado sobre os conceitos de MTs. Estas afirmações deviam ser respondidas também por meio da escala de Likert. O objetivo principal desta seção foi descobrir se a aplicação pode ser considerada uma forte aliada no aprendizado dos estudantes, ou se ainda necessita de muitas outras funcionalidades para esta finalidade. A seguir, estão listadas as 5 afirmações presentes nesta seção:

1. A ferramenta Turing Station me ajudou a entender os conceitos de Máquinas de Turing;
2. O uso da Turing Station melhorou minha capacidade de resolver problemas relacionados a Máquinas de Turing;
3. Os exemplos e simulações disponíveis na Turing Station foram claros e úteis para o

aprendizado;

4. A ferramenta permitiu uma exploração prática e intuitiva das variantes de Máquinas de Turing;
5. Eu acho que a ferramenta Turing Station pode ser uma boa ferramenta auxiliar para o ensino e aprendizado dos conceitos de Máquinas de Turing.

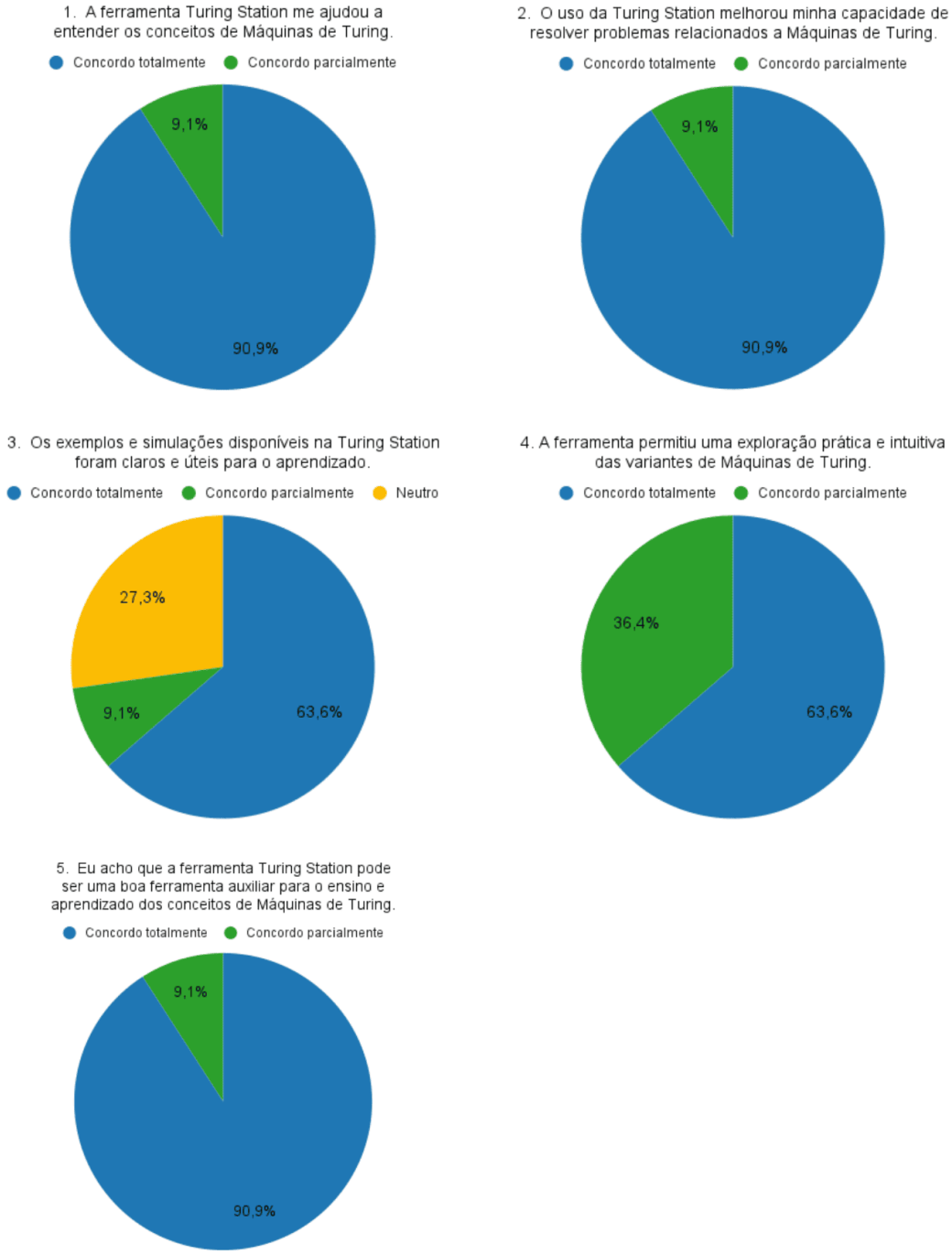
A Figura 52 apresenta os resultados obtidos para as afirmações desta seção em formato de gráficos de setores. Como pode-se observar por meio da Figura 52, a aplicação Turing Station obteve resultados bastante positivos quando se trata da sua contribuição para complementar o ensino de MTs.

Para quase todas as afirmações, os resultados se mantiveram entre Concordo parcialmente e Concordo totalmente. As respostas para as afirmações *"1. A ferramenta Turing Station me ajudou a entender os conceitos de Máquinas de Turing"* e *"2. O uso da Turing Station melhorou minha capacidade de resolver problemas relacionados a Máquinas de Turing"* demonstraram que a Turing Station contribuiu significativamente para o entendimento dos conceitos de MTs, além de ajudar na resolução dos problemas apresentados na lista de questões.

A afirmação *"3. Os exemplos e simulações disponíveis na Turing Station foram claros e úteis para o aprendizado"* obteve um intervalo de respostas maior, incluindo algumas respostas neutras, mas nenhuma negativa. Isso demonstrou que, embora a Turing Station apresente diversos exemplos para tornar o aprendizado de sua utilização mais fácil, alguns usuários afirmam implicitamente que ela ainda precisa de mais exemplos práticos para auxiliar em sua introdução.

Os resultados da afirmação *"4. A ferramenta permitiu uma exploração prática e intuitiva das variantes de Máquinas de Turing"* provaram que a Turing Station abrange de forma bem estruturada as variantes de MTs, e que a criação das mesmas acontece de forma intuitiva e prática. Já os resultados da afirmação *"5. Eu acho que a ferramenta Turing Station pode ser uma boa ferramenta auxiliar para o ensino e aprendizado dos conceitos de Máquinas de Turing"* mostraram que a aplicação tem o potencial necessário para ser uma grande aliada no entendimento das MTs e suas variantes por parte dos estudantes, e que provavelmente também possui um grande potencial para auxiliar aos professores na disciplina de Teoria da Computação.

Figura 52 – Resultados obtidos na seção de impactos educacionais da aplicação.



Fonte: Elaborada pelo autor.

8 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo central o desenvolvimento e a avaliação de uma aplicação web, denominada Turing Station, destinada a apoiar o estudo e a prática dos conceitos de Máquinas de Turing. Como metodologia para o desenvolvimento da Turing Station, o presente trabalho adotou diversas etapas de planejamento e desenvolvimento para garantir a qualidade da aplicação, tanto no quesito experiência de uso, como na sua efetividade educacional.

Inicialmente, uma análise subjetiva das características gerais, funcionalidades, e aspectos de experiência de usuário e de interface de usuário de três ferramentas com objetivos similares foi realizada. Esta análise ajudou a entender melhor as funcionalidades que uma aplicação para esta finalidade precisa possuir, além de demonstrar as limitações e dificuldades que os usuários poderiam enfrentar ao utilizar tais ferramentas. Com esta análise, foi possível definir os requisitos mínimos da Turing Station, além de permitir adicionar diversos outros requisitos que pudessem sanar algumas das limitações das aplicações similares analisadas, indicando os padrões que a Turing Station precisava atingir para que fosse considerada uma boa aplicação para o estudo de MTs.

A escolha da web como meio de utilização da aplicação se deveu ao fato dela estar presente nos mais diversos tipos de dispositivos, seja dispositivos de mesa, dispositivos móveis, televisões inteligentes, e muito mais, quebrando barreiras e democratizando o acesso as suas funcionalidades. O desenvolvimento da Turing Station foi focado em utilizar tecnologias que permitam uma grande escalabilidade de funcionalidades, otimização, e responsividade.

O *design* da Turing Station foi planejado para ser atrativo e bem estruturado, proporcionando uma boa experiência de uso e a fluidez necessária para que os usuários pudessem usufruir das suas funcionalidades. Como mostrado nas seções 7.2 e 7.3, os usuários, em sua grande maioria, acharam que a Turing Station possui uma interface de usuário bastante agradavél e uma estrutura bem organizada, consistente e intuitiva. A aplicação obteve a nota $A+$ para as respostas do questionário SUS, sendo esta a nota mais alta possível para esta metodologia de medição de usabilidade, reforçando o seu compromisso em oferecer a melhor experiência possível no aprendizado dos conceitos de MTs.

Da mesma forma, a Seção 7.4 demonstrou que a aplicação foi considerada uma boa aliada na exploração dos conceitos de MTs, com os respondentes da pesquisa realizada afirmando que a Turing Station os ajudou a entender melhor os conceitos de MTs e suas variantes, além de ter melhorado as suas capacidades de resolução de problemas relacionados. Isto mostra que a

ferramenta está no caminho certo de tornar estes conceitos tão abstratos bem mais amigáveis e inclusivos.

Como trabalhos futuros, planeja-se continuar aprimorando a aplicação, com a adição de novas funcionalidades, correção de erros, e otimizações. Como novas funcionalidades, pretende-se adicionar a possibilidade de exportação para a biblioteca Teocomp, apresentada em 4.3, possibilitar o uso de subrotinas e adicionar as funcionalidades mais importantes que foram reportadas pelos usuários no formulário de avaliação, como a funcionalidade de seleção de elementos no simulador, poder dar *zoom* com o *mousepad* e com toques na tela, adicionar mais atalhos de teclado, entre outros. Ademais, pretende-se realizar novos testes com uma quantidade maior de usuários após realizada as devidas melhorias na aplicação, para que possa se confirmar com mais precisão a efetividade da Turing Station.

REFERÊNCIAS

ALLISON, C.; MILLER, A.; OLIVER, I.; MICHAELSON, R.; TIROPANIS, T. The Web in education. **Computer Networks**, Elsevier, v. 56, n. 18, p. 3811–3824, 2012.

ALVES, D.; GUERRA, P. Uma ferramenta web para criação e simulação de máquinas de Turing. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. **Anais [...]**. Porto Alegre, RS, Brasil: SBC, 2024. p. 2301–2312. ISSN 0000-0000.

BHAT, K. **Ultimate Tailwind CSS handbook**: build sleek and modern websites with immersive uis using Tailwind CSS (english edition). [*S. l.*]: Orange Education Pvt Limited, 2023. ISBN 9789388590761.

BIØRN-HANSEN, A.; MAJCHRZAK, T. A.; GRØNLI, T.-M. Progressive web apps: the possible web-native unifier for mobile development. In: INTERNATIONAL CONFERENCE ON WEB INFORMATION SYSTEMS AND TECHNOLOGIES. **Proceedings [...]**. [*S. l.*], 2017. v. 2, p. 344–351.

BROOKE, J. **SUS**: a 'quick and dirty' usability scale. In: JORDAN, P. W.; THOMAS, B.; WEERDMEESTER, B. A.; MCCLELLAND, I. L. (Ed.). Usability Evaluation in Industry. London: Taylor & Francis, 1996. p. 189–194.

BUDDE, R.; KAUTZ, K.; KUHLENKAMP, K.; ZÜLLIGHOVEN, H. What is prototyping? **Information Technology & People**, MCB UP Ltd, v. 6, n. 2/3, p. 89–95, 1990.

CHOUDHURY, N. World wide web and its journey from web 1.0 to web 4.0. **International Journal of Computer Science and Information Technologies**, v. 5, n. 6, p. 8096–8100, 2014.

CLARK, R.; MAYER, R. **E-learning and the science of instruction**: proven guidelines for consumers and designers of multimedia learning. 5. ed. [*S. l.*]: Wiley, 2023. 3–17 p. ISBN 9781394177387.

COHEN, J. Non-deterministic algorithms. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 11, n. 2, p. 79–94, jun 1979. ISSN 0360-0300.

DEVEDŽIC, V. **Semantic web and education**. [*S. l.*]: Springer US, 2006. 1–28 p. (Integrated Series in Information Systems). ISBN 9780387354170.

ELMUNSYAH, H.; HIDAYAT, W.; ULFA, S.; SURAKHMAN, E.; WAKHIDAH, R. Measuring user experience on personalized online training system to support online learning. In: IOP CONFERENCE SERIES: MATERIALS SCIENCE AND ENGINEERING. **Proceedings [...]**. [*S. l.*], 2020. v. 732, n. 1, p. 012115.

GACKENHEIMER, C. **Introduction to React**. [*S. l.*]: Apress, 2015. (Books for professionals by professionals). ISBN 9781484212455.

GAUTAM, S.; TIWARI, M. K. Components and benefits of E-learning system. **International Research Journal of Computer Science (IRJCS)**, v. 3, n. 1, p. 14–17, 2016.

GUNTUPALLI, R. C. C. **User Interface design**: methods and qualities of a good user interface design. 42 p. Dissertação (Mestrado em Engenharia de Software) – University West, Trollhättan, Sweden, 2008.

HALEEM, A.; JAVAID, M.; QADRI, M. A.; SUMAN, R. Understanding the role of digital technologies in education: a review. **Sustainable Operations and Computers**, Elsevier, v. 3, p. 275–285, 2022. ISSN 2666-4127.

HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. **Introduction to automata theory, languages, and computation**. 3. ed. USA: Addison-Wesley Longman Publishing Co., Inc., 2006. 324–334, 341–349 p. ISBN 0321455363.

HUME, D. **Progressive Web Apps**. [*S. l.*]: Simon and Schuster, 2017. ISBN 9781638351412.

LEWIS, J. R. The system usability scale: past, present, and future. **International Journal of Human–Computer Interaction**, Taylor & Francis, v. 34, n. 7, p. 577–590, 2018.

LIKERT, R. A technique for the measurement of attitudes. **Archives of Psychology**, v. 22 140, p. 55–55, 1932.

LINZ, P.; RODGER, S. **An introduction to formal languages and automata**. 7. ed. [*S. l.*]: Jones & Bartlett Learning, 2022. 247–292 p. ISBN 9781284263282.

LOSACCO, M.; RODGER, S. FLAP: a tool for drawing and simulating automata. **Media**, ERIC, v. 93, p. 310–317, 1993.

MCCARTHY, L.; REAS, C.; FRY, B. **Getting started with p5.js**: making interactive graphics in JavaScript and Processing. [*S. l.*]: Make Community, LLC, 2015. 1–3 p. ISBN 9781457186738.

MEYER, E.; WEYL, E. **CSS**: the definitive guide. 4. ed. [*S. l.*]: O'Reilly Media, 2017. 1–2 p. ISBN 9781449325091.

PEREIRA, C. **Aplicações web real-time com Node.js**. [*S. l.*]: Casa do Código, 2014. 1–3 p. ISBN 9788566250930.

PEREIRA, C. H.; TERRA, R. A mobile app for teaching formal languages and automata. **Computer Applications in Engineering Education**, v. 26, n. 5, p. 1742–1752, 2018.

PROCOPIUC, M.; PROCOPIUC, O.; RODGER, S. Visualization and interaction in the computer science formal languages course with JFLAP. In: TECHNOLOGY-BASED RE-ENGINEERING ENGINEERING EDUCATION PROCEEDINGS OF FRONTIERS IN EDUCATION FIE'96 26TH ANNUAL CONFERENCE. **Proceedings [...]**. [*S. l.*], 1996. v. 1, p. 121–125 vol.1.

RODGER, S. H.; WIEBE, E.; LEE, K. M.; MORGAN, C.; OMAR, K.; SU, J. Increasing engagement in automata theory with JFLAP. In: PROCEEDINGS OF THE 40TH ACM TECHNICAL SYMPOSIUM ON COMPUTER SCIENCE EDUCATION. **Proceedings [...]**. [*S. l.*], 2009. p. 403–407.

SAURO, J.; LEWIS, J. **Quantifying the User Experience**: practical statistics for user research. [*S. l.*]: Morgan Kaufmann, 2016. ISBN 9780128025482.

SILVA, L.; DIAS, B.; FINGER, A.; SILVA, W. Avaliação da experiência de uso do JFLAP como recurso pedagógico no ensino de linguagens formais. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. **Anais [...]**. Porto Alegre, RS, Brasil: SBC, 2023. p. 995–1006. ISSN 0000-0000.

SILVA, M. **JavaScript - guia do programador**: guia completo das funcionalidades de linguagem JavaScript. [*S. l.*]: Novatec Editora, 2010. 22–23 p. ISBN 9788575222485.

SILVA, M. **HTML5**: a linguagem de marcação que revolucionou a web. 2. ed. [*S. l.*]: Novatec Editora, 2019. 20–26 p. ISBN 9788575228029.

SIPSER, M. **Introduction to the Theory of Computation**. 3. ed. Boston: Cengage Learning, 2012. 165–182 p. ISBN 9781133187813.

SOMMERVILLE, I. **Software Engineering**. 10. ed. [*S. l.*]: Pearson, 2015. 102–111 p. ISBN 0133943038.

TURING, A. On computable numbers, with an application to the Entscheidungsproblem. **Proceedings of the London Mathematical Society**, Association for Symbolic Logic, v. 42, n. 1, p. 230–265, 1936.

VASCONCELOS, D.; GUERRA, P. T. Ensinando teoria da computação com Jupyter Notebook. In: WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO. **Anais [...]**. Porto Alegre, RS, Brasil: SBC, 2023. p. 9–19. ISSN 2595-6175.

VESSELINOV, R.; GREGO, J. **Duolingo effectiveness study**. City University of New York, USA, v. 28, n. 1-25, 2012.

WARD, C. **Jump start responsive web design**: modern responsive solutions. 2. ed. [*S. l.*]: SitePoint, 2017. ISBN 9781492020639.

APÊNDICE A – QUESTÕES DO FORMULÁRIO DE AVALIAÇÃO DA TURING STATION

A.1 Seção 1: Termo de Consentimento Livre e Esclarecido (TCLE)

1. Eu declaro que é de livre e espontânea vontade minha participação nesta pesquisa.
2. Eu declaro que li e compreendi todas as informações apresentadas neste Termo de Consentimento Livre e Esclarecido e estou de acordo em participar.

A.2 Seção 2: Questões Demográficas

1. Curso
2. Ano de ingresso

A.3 Seção 3: Questões relacionadas a Experiência de Uso na plataforma Turing Station

1. Eu acho que gostaria de usar esse sistema com frequência.
2. Eu acho o sistema desnecessariamente complexo.
3. Eu achei o sistema fácil de usar.
4. Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.
5. Eu acho que as várias funções do sistema estão muito bem integradas.
6. Eu acho que o sistema apresenta muita inconsistência.
7. Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente.
8. Eu achei o sistema atrapalhado de usar.
9. Eu me senti confiante ao usar o sistema.
10. Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.

A.4 Seção 4: Questões relacionadas a Experiência de Usuário (UX) e Interface de Usuário (UI) na Turing Station

1. Em qual dispositivo você utilizou a Turing Station?
2. Eu acho que o design da interface da Turing Station contribuiu para uma experiência mais agradável.

3. Eu acho que as instruções fornecidas pela ferramenta são claras e suficientes para iniciar e realizar simulações de Máquinas de Turing.
4. Eu acho que navegar entre as funcionalidades da Turing Station foi intuitivo.
5. Eu acho que os tempos de resposta e a estabilidade da Turing Station atenderam às expectativas.
6. O que você mais gostou na Turing Station?
7. Há algo que poderia ser melhorado para tornar a Turing Station mais útil ou intuitiva? Se sim, o que seria?

A.5 Seção 5: Questões relacionadas ao impacto da Turing Station no aprendizado

1. A ferramenta Turing Station me ajudou a entender os conceitos de Máquinas de Turing.
2. O uso da Turing Station melhorou minha capacidade de resolver problemas relacionados a Máquinas de Turing.
3. Os exemplos e simulações disponíveis na Turing Station foram claros e úteis para o aprendizado.
4. A ferramenta permitiu uma exploração prática e intuitiva das variantes de Máquinas de Turing.
5. Eu acho que a ferramenta Turing Station pode ser uma boa ferramenta auxiliar para o ensino e aprendizado dos conceitos de Máquinas de Turing.

APÊNDICE B – RESPOSTAS PARA O ITEM 6 DA SEÇÃO 4 DO FORMULÁRIO DE AVALIAÇÃO DA TURING STATION

Abaixo estão listadas as respostas adicionadas pelos os usuários para o item *"6. O que você mais gostou na Turing Station?"* no formulário de avaliação da Turing Station.

- "Gostei da simplicidade de se criar estados, liga-los entre si e poder realizar testes que motram o passo-a-passo da máquina de Turing."
- "Da possibilidade de mover o canvas, pois, assim, é possível por porções da máquina de turing na área de clipagem e depois mover o canvas novamente para visualizar essas porções. Isso faz com que seja possível a criação de máquinas de turing com muitos estados."
- "Poder simular passo a passo minhas MTs."
- "Poder ver os passo-a-passo da configuração da máquina, multiteste, poder ficar com várias máquinas abertas e design bonito."
- "Da facilidade de criar as máquinas e a facilidade de fazer modificações nela."
- "O que mais gostei foi a facilidade e velocidade de se testar as entradas da máquina de turing, a interface do sistema também é muito agradável e intuitiva."
- "A interface é muito intuitiva e simples, e uma das coisas que definitivamente mais me agradou foi o fato de não precisar instalar nada para fazer uso dela e também usar ela no celular que foi muito útil pra situações que o notebook não estava disponível."
- "A plataforma é muito bem-feita. Parabéns. A facilidade e integração com os conteúdos vistos em sala de aula, estão ótimas."
- "Acredito que a UI do Turing Station é bem mais polida e intuitiva do que outras ferramentas que tem o mesmo propósito."

APÊNDICE C – RESPOSTAS PARA O ITEM 7 DA SEÇÃO 4 DO FORMULÁRIO DE AVALIAÇÃO DA TURING STATION

Abaixo estão listadas as respostas adicionadas pelos os usuários para o item *"7. Há algo que poderia ser melhorado para tornar a Turing Station mais útil ou intuitivo? Se sim, o que seria?"* no formulário de avaliação da Turing Station.

- "Seria muito interessante adicionar a funcionalidade de seleção (muito comum em softwares gráficos e ajudaria bastante a trabalhar com múltiplos estados) e melhorar o histórico de edição da máquina de Turing (atualmente só está sendo possivel desfazer o último passo)."
- "Seria melhor se, ao dar 'zoom in' ou 'zoom out', só as proporções do canvas se alterassem e não o canvas e o menu lateral."
- "Sim, quando eu atingi um certo número de MTs criadas, o site começou a ficar lento. Para melhorar a usabilidade, seria interessante permitir que o usuário utilizasse atalhos de teclado, como pressionar a tecla 'O' para criar um estado, algo parecido com o Figma."
- "Dar zoom com mousepad, ctrl z e y, poder ficar com os testes aparecendo enquando manipula os estados da máquina, exportação da imagem, poder importar uma máquina com outras abertas, não poder selecionar vários estados com suas conexões e copia-los e colar-los..."
- "Algo que tenho a comentar é sobre a exportação de imagens PNG, não sei se foi algo no meu PC, mas a qualidade da imagem saiu péssima."
- "Existem alguns problemas de lag (configurações da máquina, caso necessário para benchmark: i5 12th, 24GB ram, RTX 3050) e crashes inesperados, isso, quando temos muitas máquinas de turing criadas. Além disso, seria uma boa adição para o usuário, deixá-lo editar a variante da máquina de turing após criada. (i.e. Havia começado uma MT, mas me dei conta que precisava ser infinita à esquerda. Como não conseguia editar, tive que refazê-la).Consertar o bug de que não é possível importar uma máquina de turing individual; o bug onde, se uma instrução causa um loop onde a agulha fica indo e voltando, a máquina continua rodando mesmo que eu clique em pausar."