Um sistema para auxiliar na aprendizagem da disciplina Linguagens Formais e Autômatos

Rafael Cardoso da Silva, Daniel Morgato Martin
Centro de Matemática, Computação e Cognição, Universidade Federal do ABC
Av. dos Estados, 5001, Santo André, SP
rafael.cardoso@aluno.ufabc.edu.br, daniel.martin@ufabc.edu.br

Resumo: Resolver exercícios é fundamental para um aluno fixar os conceitos apresentados em aula. Por outro lado, ter seus exercícios corrigidos também é muito importante, para que ele possa avaliar o seu aprendizado. Na UFABC, a disciplina de Linguagens Formais e Autômatos contempla vários exercícios que admitem infinitas respostas, o que torna a correção deles praticamente impossível, principalmente quando as turmas são grandes. O objetivo deste projeto foi a criação e implementação de um sistema para aplicação e correção automática de exercícios envolvendo autômatos finitos determinísticos. Através do estudo de métodos e algoritmos presentes na literatura, foi possível implementar o teste de equivalência entre o autômato-resposta do aluno e o autômato-gabarito previamente armazenado no banco de dados. Ao final do projeto, o sistema foi usado em caráter experimental numa turma da UFABC da disciplina de Linguagens Formais e Autômatos, afim de testar a sua qualidade. E ao final da disciplina, a nota que os alunos obtiverem ao revolver os exercícios do sistema ajudarão a compor o conceito final de cada um na disciplina.

Palavras-chave: autômato finito, equivalência de autômatos, minimização de autômato, programação para web.

Introdução

Um *autômato finito determinístico* (também chamado de AFD ou máquina de estados) consiste de um conjunto de estados não vazio Q, um alfabeto Σ, um estado inicial $s \in Q$, um conjunto de estados de aceitação $F \subseteq Q$ e uma função de transição $\delta\colon Q \times \Sigma \to Q$ (HOPCROFT; MOTWANI; ULLMAN, 2006).

A seguir, dois exemplos de AFDs que reconhecer a linguagem $\{w \in \Sigma^* : w$ tem número par de símbolos $\mathtt{a}\}$.

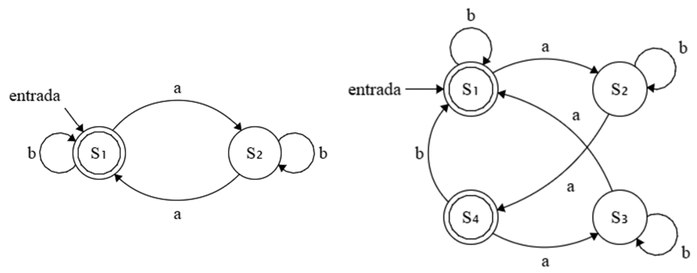

Figura 1: *Autômato M_1 a esquerda e Autômato M_2 a direita.*

Apesar de M_2 ser visivelmente maior e mais complexo que M_1 da Figura 1, ambos reconhecem a mesma linguagem. É possível demonstrar que, para cada linguagem regular, existem infinitos autômatos que a reconhecem. Como decidir então se o autômato-resposta dado por um aluno reconhece a linguagem pedida?

No caso do sistema que iremos desenvolver isso se reduz ao seguinte problema:

Problema: Decidir se dois autômatos finitos determinísticos reconhecem a mesma linguagem.

Objetivos e Metodologia

- Estudar e implementar o algoritmo de Moore para minimização e equivalência de AFDs (MOORE, 1956);
- Estudar e implementar o algoritmo de Hopcroft e Karp para testar a equivalência de AFDs (HOPCROFT; KARP, 1971);
- Desenvolvimento do sistema de apoio a aprendizagem da disciplina Linguagens Formais e Autômatos;
- Reuniões semanais com o orientador, afim de acompanhar o progresso do desenvolvimento do sistema e esclarecer eventuais dúvidas.

O Juiz do Sistema

Algoritmo 1: Teste de Equivalência de Hopcroft e Karp.

Entrada: $Q, \Sigma, \delta, \{p_0, q_0\}$
Saída: Conjuntos disjuntos
1 **início**
2 conjuntos.INIT($|Q|$)
3 conjuntos.MERGE(p_0, q_0)
4 pilha.empilha($\{p_0, q_0\}$)
5 **enquanto** *pilha* $\neq \emptyset$ **faça**
6 $\{p, q\} \leftarrow$ pilha.desempilha()
7 **para cada** $\sigma \in \Sigma$ **faça**
8 $r \leftarrow$ conjuntos.FIND($\delta(p, \sigma)$)
9 $s \leftarrow$ conjuntos.FIND($\delta(q, \sigma)$)
10 **se** $r \neq s$ **então**
11 conjuntos.MERGE(r, s)
12 pilha.empilha($\{r, s\}$)
13 **retorna** *conjuntos*

Modelagem do Sistema

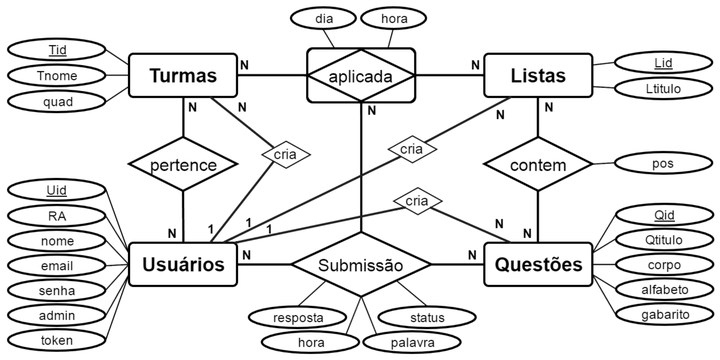

Figura 2: *Diagrama de Entidade e Relacionamento do DFAjudge*

Tipos de Feedback

- Correto;
- Incorreto, e uma palavra que os distinguiram;
- Salvo como Rascunho.

Interface do DFAjudge

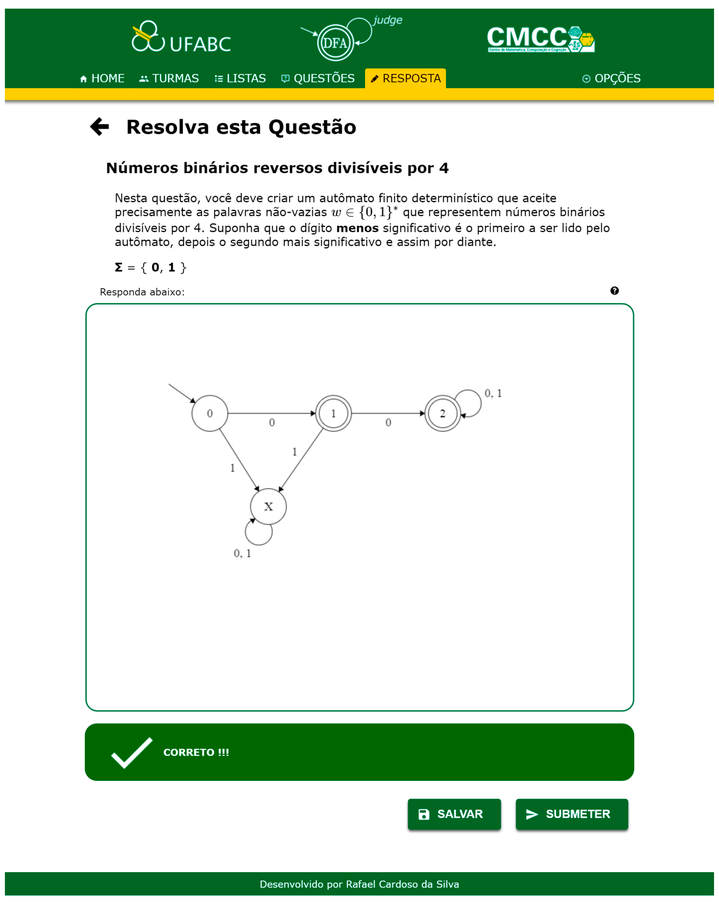

Figura 3: *Fomulário para Resolver uma Questão*

Futuros Projetos

- Cobrir outras classes de exercícios da disciplina de Linguagens Formais e Autômatos, e também de outras disciplinas, desde que admitem algoritmos corretores;
- Aperfeiçoar o gerenciamento de Questões;
- Aprimorar o DFAdesigner;
- Ser aberto ao público.

Referências

HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. *Automata theory, languages, and computation*. International Edition, v. 24, 2006.

MOORE, Edward F. *Gedanken-experiments on sequential machines*. Automata studies, v. 34, p. 129-153, 1956.

HOPCROFT, John E.; KARP, Richard M. *A Linear Algorithm for Testing Equivalence of Finite Automata*. Technical report of Cornell University, p. 71–114, 1971.

Agradecimentos

Este trabalho foi financiado pelo Programa de Iniciação Científica da UFABC.

Apoio
CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico
UFABC
VI ENCONTRO DE INICIAÇÃO CIENTÍFICA
IX SIMPÓSIO DE INICIAÇÃO CIENTÍFICA DA UFABC
11º CONGRESSO DE INICIAÇÃO CIENTÍFICA DA USCS