Um software para auxílio a aprendizagem de Linguagens Regulares

Marlon José Dognini *dognini@uol.com.br*
André Luís Alice Raabe *araabe@cttmar.univali.br*

Ciência da Computação - CTTMar - Universidade do Vale do Itajaí
Rua Uruguai, 458 - Bloco 7 – 88.302-202 – Itajaí – SC - Brasil
Telefone (0xx47) 341-7544

Resumo

Este trabalho apresenta o desenvolvimento do EduLing: software educacional para auxilio a aprendizagem de Linguagens Regulares, um tema pertinente ao estudo das Linguagens Formais. O software possibilita ao aluno experimentar a construção de autômatos e expressões regulares através de uma interface gráfica interativa. O aluno pode verificar a equivalência entre as diferentes representações de uma linguagem regular, bem como simular o reconhecimento de sentenças a fim de validar a linguagem construída. Resultados preliminares indicam que o software contribui para melhoria da qualidade do Ensino de Ciência da Computação.

Palavras-Chave*: Ambientes Interativos de Aprendizagem; Linguagens Regulares; Software Educacional; Linguagens Formais.*

Abstract

This paper presents the development of EduLing, an educational software for aid the learning of Regular Languages. The software allows students to experiment automatas and regular expressions construction and to verify the equivalence among them. EduLing interface was designed to help students to switch among different representations of regular languages and also simulate sentence recognition. Preliminary results indicate that the software help to increase the quality of Computes Science teaching.

Key words*: Interactive Learning Environments; Regular Languages; Educational Software; Formal Languages.*

1. Introdução

O estudo das linguagens formais é um conteúdo obrigatório na grade curricular da maioria dos cursos de Ciência da Computação do Brasil. A disciplina de Linguagens Formais normalmente aborda assuntos como conjuntos, alfabetos, linguagens regulares, autômatos, gramáticas livres de contexto, sensíveis ao contexto e irrestritas, máquinas de pilha, máquinas de Turing e Post, decidibilidade e assim por diante. Tais conteúdos são essenciais para o desenvolvimento do aluno no curso de Ciência da Computação, e dão suporte a disciplinas como Compiladores e Inteligência Artificial.

Tradicionalmente a disciplina possui uma característica formal, a qual os alunos apresentam dificuldade para aprendizado devido ao grau de abstração exigido. Geralmente a dificuldade inicia pela falta de familiaridade com a representação formal.

Com isso, a tarefa do professor de Linguagens Formais, normalmente se torna repleta de ilustrações e exemplos utilizando representações gráficas (especialmente nos autômatos), longas cadeias de caracteres, simulações da geração e reconhecimento de sentenças entre outras.

Uma das etapas iniciais da disciplina aborda as linguagens regulares, as quais podem ser representadas através de Autômatos Finitos Determinísticos (AFD), Autômatos Finitos Não-Determinísticos (AFND), Tabelas de Transição e Expressões Regulares (ER). Por ser um conceito inicial e que normalmente leva o aluno a principiar no processo de adequação a notação e aos processos formais, acredita-se que seja o momento ideal de utilizar uma abordagem alternativa proporcionada pelo uso de um *software* educacional.

Tendo em vista esta realidade, foi desenvolvida uma ferramenta para auxiliar no processo de ensino-aprendizagem da disciplina de Linguagens Formais, mais especificamente sobre as linguagens regulares. O EduLing possibilita o desenvolvimento de atividades práticas sobre a construção de linguagens regulares, seja através de Autômatos Finitos Determinísticos (AFD), Não-Determinístico (AFND), expressões regulares ou tabelas de transição.

O *software* auxilia na ilustração dos processos que demonstram a equivalência entre as representações (ER, AFD, AFND) e também possibilita a experimentação das hipóteses de estudo dos alunos com relação a construção de linguagens regulares e de reconhecedores para estas.

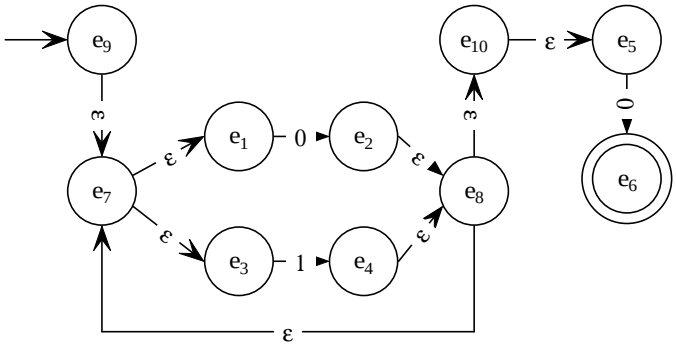
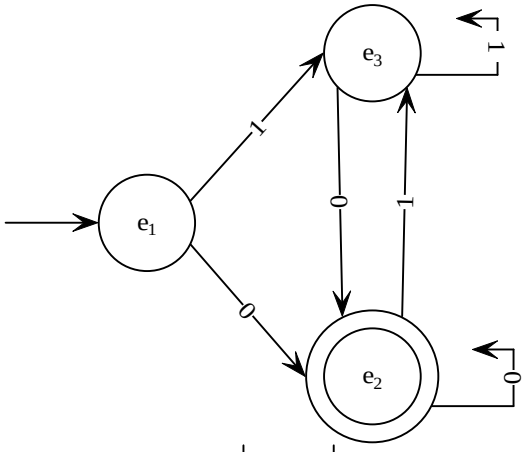
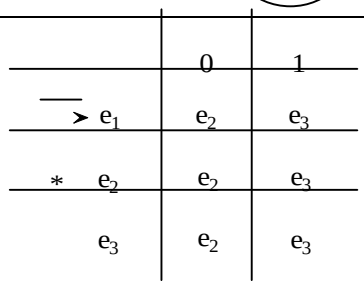
Este artigo está organizado da seguinte forma: a seção 2 apresenta os conceitos de linguagens regulares e equivalências tratados pelo EduLing; a seção 3 detalha o EduLing e suas funcionalidades; a seção 4 apresenta as concepções pedagógicas que nortearam o desenvolvimento do software; a seção 5 apresenta os resultados preliminares e a seção 6 as considerações finais e perspectivas futuras deste trabalho.

2. Linguagens Regulares

As linguagens artificiais apresentam-se em classes hierárquicas conforme definido por Chomsky (1955). Sua classificação divide-se em: linguagens regulares, livres de contexto, sensíveis ao contexto e irrestritas. De acordo com a Hierarquia de Chomsky, as linguagens regulares são a classe de linguagens mais simples, possibilitando o desenvolvimento de algoritmos de pouca complexidade, grande eficiência e de fácil implementação [Rozenberg & Salomaa, 1997].

Uma linguagem regular pode ser representada por Expressão Regular (ER), Autômato Finito Determinístico (AFD), Autômato Finito Não Determinístico (AFND) e tabela de transição. A Tabela 1 descreve brevemente cada uma dessas representações e apresenta um exemplo de cada uma delas.

Tabela 1. Representações de uma linguagem regular

Descritivo	Exemplo
Expressão Regular denota um formalismo para construir palavras de uma linguagem, definida a partir de um conjunto básico de operações de concatenação e união [Menezes, 1997].	(0 | 1) * 0
Autômato Finito Não Determinístico apresenta duas ou mais transições para fora de um estado, possibilitando o símbolo de entrada seguir por qualquer dos "caminhos", este por sua vez, torna-se mais lento, pois é mais suscetível a erros, porém ele ocupa menor espaço de memória [Crespo, 1998].	
Autômato Finito Determinístico apresenta apenas uma única transição para fora de um estado que está representando o símbolo de entrada no mesmo, tornando-se mais rápido [Crespo, 1998].	
Tabela de transição é a representação tabular do diagrama de transição (AFD ou AFND) [Menezes, 1997].	

2.1.Equivalências entre as representações

A equivalência entre as diferentes representações de uma linguagem regular pode ser verificada a partir da aplicação de algoritmos de conversão de uma notação em outra. A Figura 1 ilustra os algoritmos aplicados pelo EduLing para demonstração da equivalência.

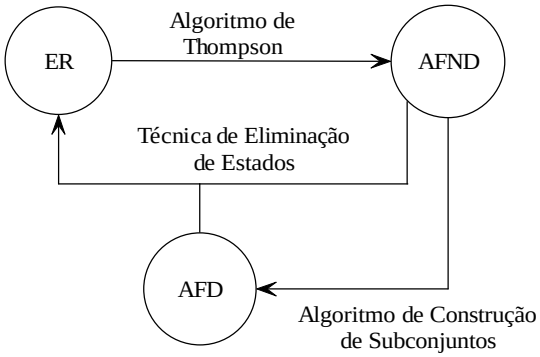


Figura 1. Algoritmos de conversão utilizados

Um maior aprofundamento sobre os algoritmos de conversão pode ser encontrado em [HOPCROFT & ULLMAN, 1979], [AHO et al., 1995], [ROZENBERG & SALOMAA, 1997].

3. Descrição do EduLing

O EduLing foi concebido para auxiliar a aprendizagem de linguagens regulares, em especial facilitando e proporcionando alternativas visuais para a construção de expressões regulares e autômatos. Para isso, foi desenvolvida uma interface interativa onde o aluno pode manipular diretamente símbolos, transições e estados, bem como simular o reconhecimento de sentenças de forma gráfica.

O EduLing está dividido em três módulos, Tutorial, Experimentação Livre e Desafio. O Módulo de Experimentação Livre está subdividido em Expressão Regular e Autômato Finito, assim o acesso a estes módulos pode ser feito através dos atalhos na tela principal do EduLing, conforme a Figura 2.

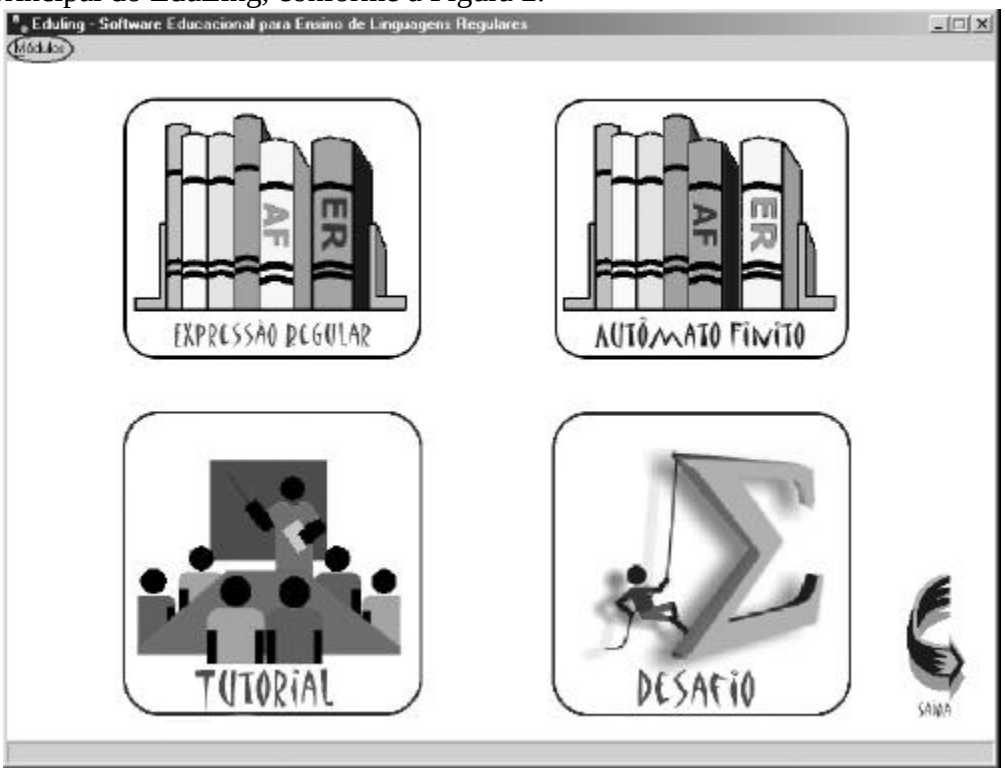


Figura 2. Tela principal do EduLing

3.1. Módulo Tutorial

O módulo tutorial aborda os conceitos fundamentais sobre a teoria das Linguagens Regulares que são utilizados pelo EduLing. Este apresenta um menu de acesso aos seis capítulos disponíveis: Expressão Regular, Autômato Finito, Autômato Finito Não Determinístico, Autômato Finito Determinístico, Minimização de Autômatos e Algoritmos de Conversão.

Cada capítulo é composto de textos, exemplos ilustrados e exercícios de múltipla escolha, onde o aluno pode verificar a compreensão que obteve do capítulo.

3.2. Módulo de Experimentação Livre

O módulo de Experimentação Livre possibilita a construção de linguagens regulares de forma prática. O aluno pode desenvolver sua linguagem através de Expressão Regular ou Diagrama de Transição. Em ambos os casos, após terminada a construção da linguagem, é possível visualizar as outras representações equivalentes (AFD, AFND, ER e Tabela) conforme ilustra a figura 3.

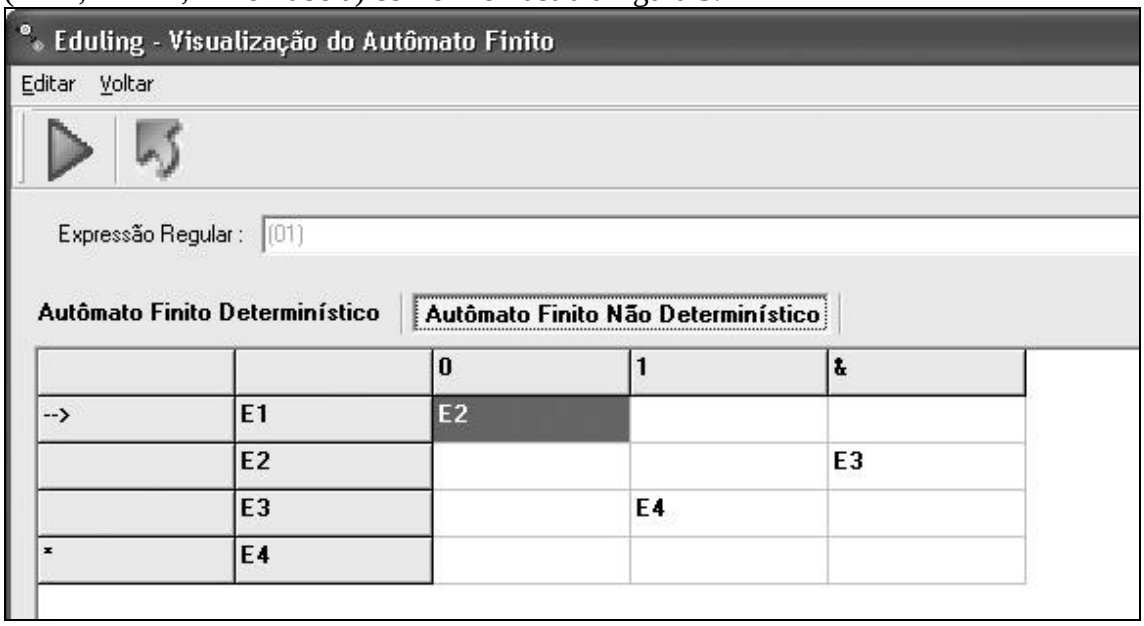


Figura 3. Ilustração da Equivalência de ER, AFD e AFND

Se o aluno optar por construir a ER ele deve fazê-lo na área de texto destinada para isso. Já se optar pela construção do Diagrama de Transição o software fornece ao aluno as ferramentas de desenho dos estados e dos arcos conforme ilustra a figura 4.

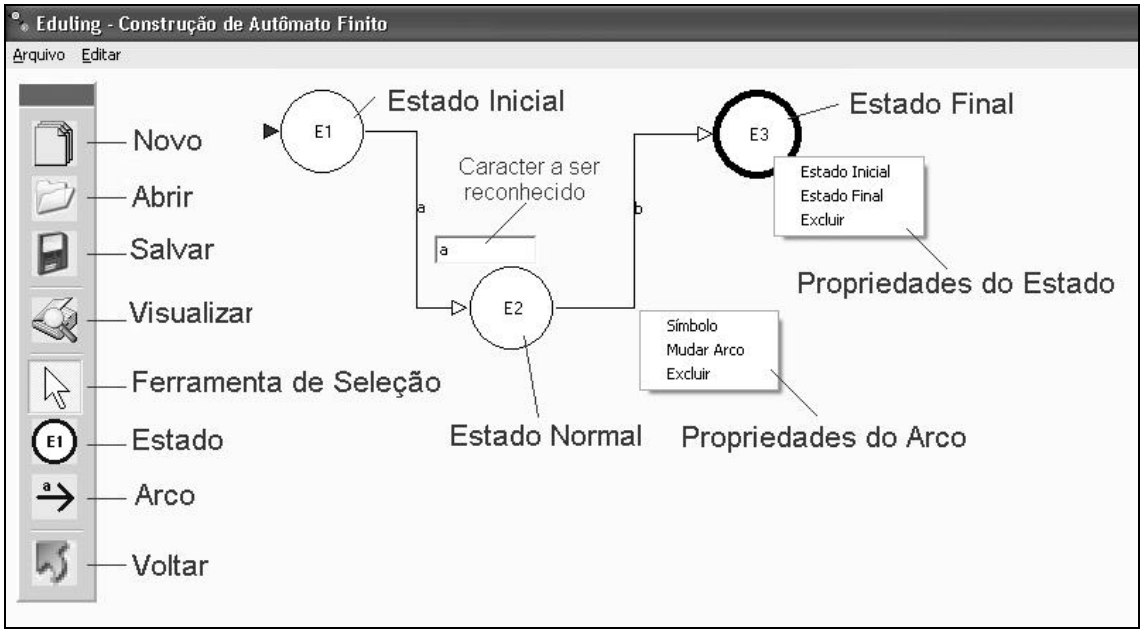


Figura 4. Construção de Diagramas de Transição

Cada estado apresenta suas propriedades, podendo ser acessadas através do botão direito do mouse, nesta operação o aluno pode marcar ou desmarcar um estado como inicial ou final. Pode também excluí-lo.

Da mesma forma, as propriedades pertinentes ao arco são acessadas com o botão direito. O aluno tem a possibilidade de informar o caractere a ser reconhecido pelo arco, modificar a localização deste (posição da ligação entre os estados) e excluí-lo.

Após o desenvolvimento do diagrama de transição, o aluno tem a possibilidade de validar a linguagem construída por meio da simulação do reconhecimento de sentenças. Para isso o aluno deve fornecer uma sentença (conjunto de caracteres) a ser reconhecida pela linguagem.

O software ilustra a simulação da seguinte forma: cada caracter da sentença de entrada é consumido disparando uma transição, neste momento, salienta-se o arco correspondente; cada mudança de estado é destacada no diagrama através da mudança de cor deste; todas as etapas da simulação ficam registradas em um log visível ao aluno. Este pode selecionar se deseja acompanhar a simulação passo a passo ou continuamente. A Figura 5 apresenta a simulação de uma sentença informada pelo aluno.

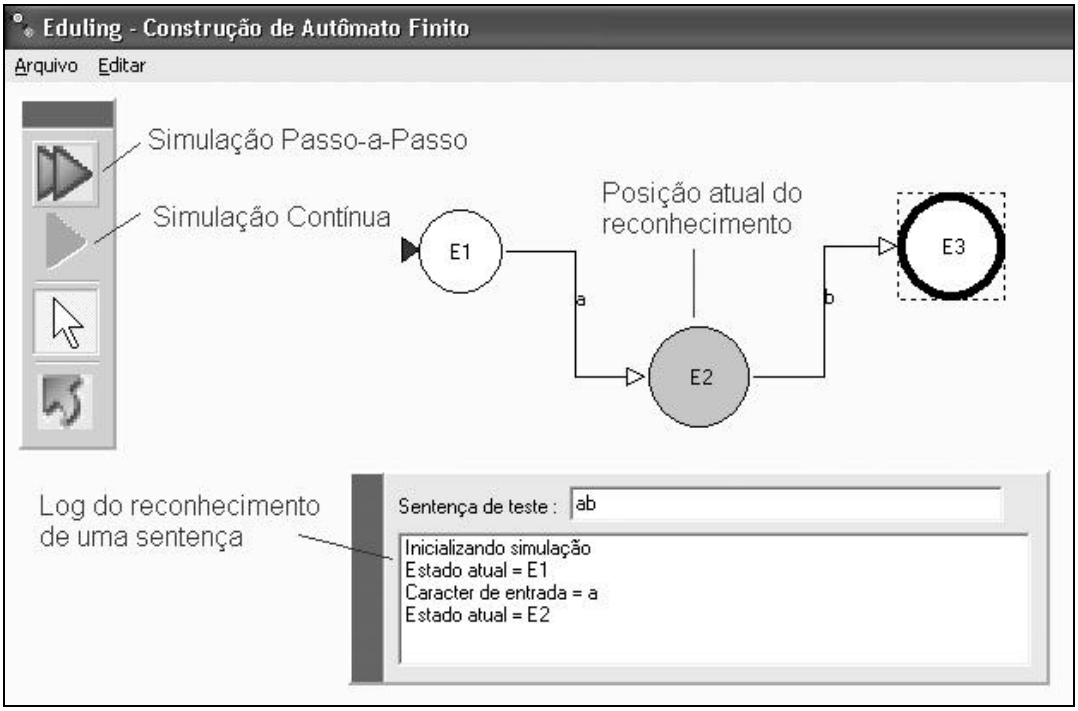


Figura 5. Simulação do diagrama desenvolvido

3.3. Módulo Desafio

O Módulo Desafio busca incentivar o aluno a explorar seu potencial criativo para resolução de problemas instigando a liberdade de expressão e aprimorando o seu senso crítico.

Neste Módulo, o professor deve anteriormente inserir um conjunto de questões a serem apresentadas ao aluno na forma de desafios. Conforme a Figura 6, o professor descreve o enunciado informando toda a dimensão do problema, indica como o aluno deverá resolver o mesmo (através de ER ou Diagrama de Transição) e elabora a resposta de acordo com a opção de resolução selecionada.

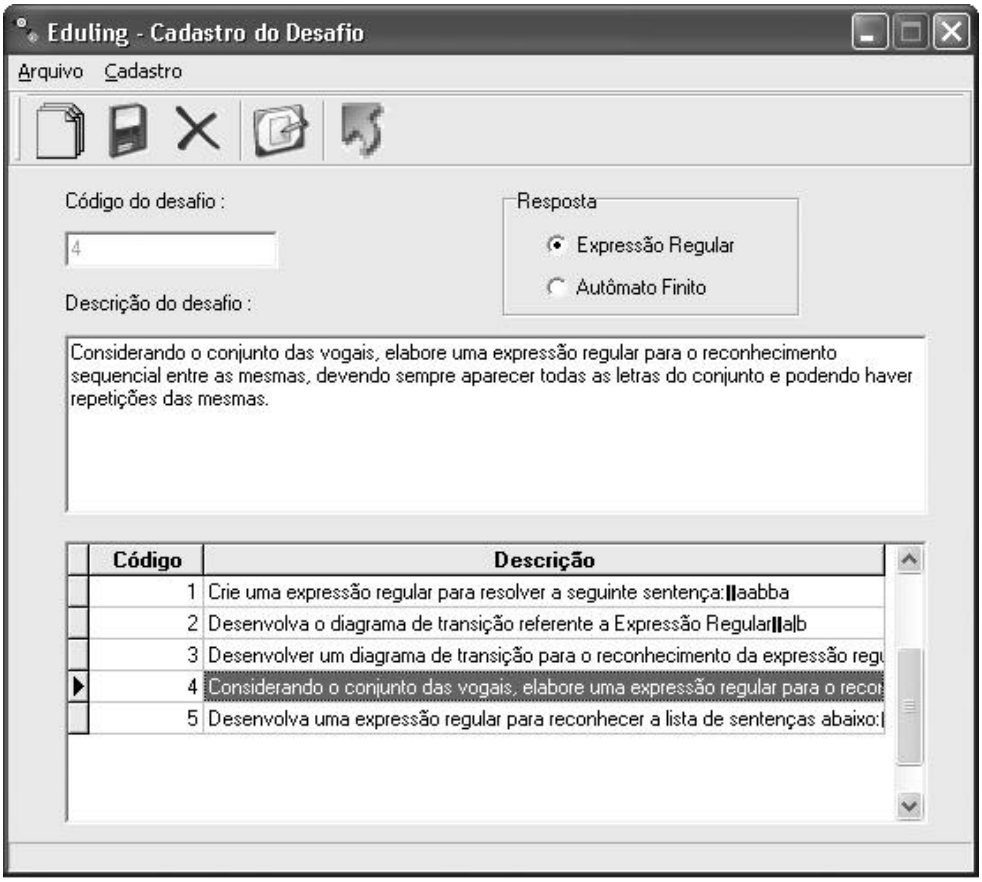


Figura 6. Inserção de um Desafio

Na execução do desafio, um problema é selecionado aleatoriamente na base de dados e apresentado ao aluno. Este inicia a resolução através do botão **Executar**. O processo de construção da solução segue o mesmo padrão do módulo de experimentação livre.

Uma vez que o aluno considere o problema solucionado, pode solicitar a avaliação do professor. Este por sua vez pode orientar o aluno a rever o processo ou mesmo exibir a sua solução acessível através do botão “Resposta” cujo acesso é restrito por uma senha. A Figura 7 mostra a tela de execução do desafio, onde o aluno tem a descrição da atividade que deverá realizar.

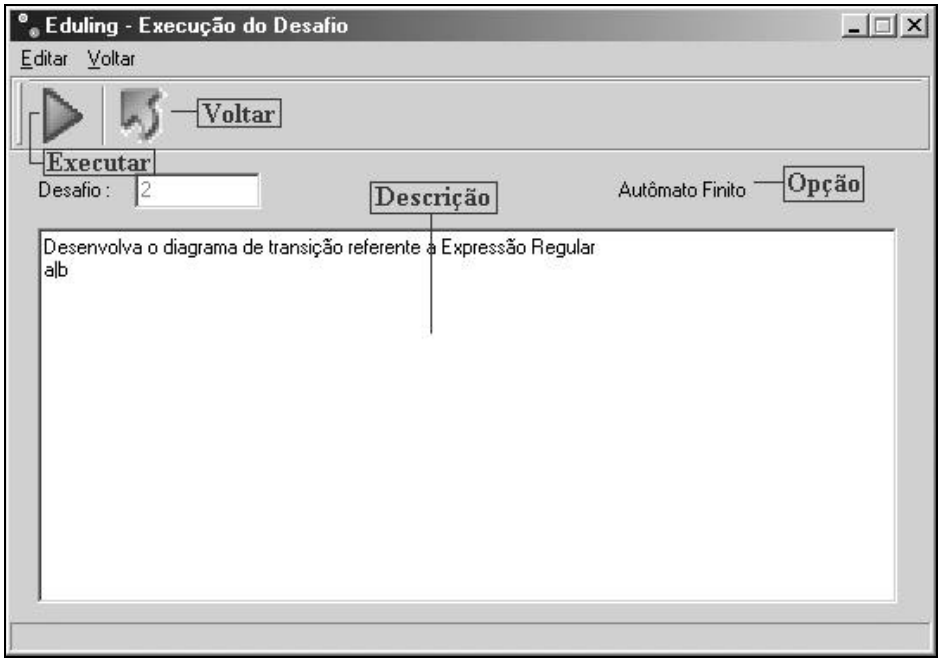


Figura 7. Tela da execução do desafio

Em qualquer momento o aluno pode salvar o trabalho que está desenvolvendo, para recuperá-lo posteriormente. O software foi desenvolvido em Delphi 5.0 e utiliza o banco de dados Paradox.

4. Concepções Pedagógicas do EduLing

O desenvolvimento de um software educacional deve considerar uma série de requisitos pertinentes ao desenvolvimento de atividades pedagógicas. A escolha de concepções pedagógicas faz-se necessária uma vez que estas deverão direcionar o desenvolvimento de todo processo de interação do software com alunos e professores.

Inicialmente, toda prática pedagógica reflete uma concepção do que seja ensinar e aprender, porém idéias e práticas acumuladas ao longo da formação de um profissional da área da educação resultam no processo de ensino-aprendizagem. Todas as decisões tomadas para a condução do trabalho pedagógico são as concepções pedagógicas [Oliveira et al., 2001].

O EduLing baseou-se em duas concepções pedagógicas distintas, porém não excludentes. O módulo de Experimentação Livre foi desenvolvido segundo a concepção racionalista e o módulo Desafio segundo a concepção interacionista. A seguir são apresentadas as concepções e como elas influenciaram no desenvolvimento da interação com o aluno no EduLing.

4.1. Concepção Racionalista

Muitos Softwares Educacionais apresentam como característica principal à concepção racionalista. Estes deixam o usuário buscar as soluções para os erros que cometeu, sem oferecer uma pista ou idéia de como solucionar tal dificuldade, assim o conhecimento deverá se dar através de simples descobertas [Oliveira et al., 2001].

No módulo de experimentação livre as atividades não são direcionadas, os recursos do software são apresentados e o aluno tem a possibilidade de explorá-los de

acordo com suas curiosidade. O software apresenta os erros do autômato construído ao aluno, validando o mesmo quanto a minimização e inconsistências. Porém o aluno não recebe nenhuma orientação para solucionar o problema, e simplesmente deve fazê-lo de forma autônoma.

4.2. Concepção Interacionista

Na concepção interacionista, o aluno não é induzido nas suas conclusões, pois a interferência do professor no processo de aprendizagem deverá ser sutil, restringindo-se apenas a orientar o aprendiz durante o desenvolvimento da solução para as fontes de pesquisa e exemplos relacionados que possam auxiliar na aprendizagem. Desta forma, o aluno desenvolve sua capacidade de reflexão e avaliação sobre o resultado, conseqüentemente fortalecendo na busca de novas alternativas para superar dificuldades futuras [Oliveira et al., 2001].

O módulo desafio foi planejado para ser desenvolvido em laboratório com a mediação do professor. Este cria anteriormente os problemas que irão desafiar o aluno de acordo com a pertinência do conteúdo a ser trabalhado. O aluno tem a possibilidade de solucioná-las com o apoio do professor, que deve intervir quando for apropriado.

Neste caso, a ferramenta facilita a atenção individualizada a cada aluno, pois cada qual estará solucionando um problema diferente. Além disso, o professor não necessita demonstrar as falhas no reconhecimento das sentenças, basta indicá-las para que o próprio aluno teste-as através da simulação.

5. Resultados Preliminares

Foi realizado um experimento preliminar com 19 alunos da disciplina de Linguagens Formais e Compiladores, para avaliar o potencial pedagógico da ferramenta no auxílio a aprendizagem de linguagens regulares. O experimento foi realizado quando os aluno estavam principiando no conteúdo relativo as linguagens regulares. O experimento foi realizado em três etapas: reconhecimento do software, experimentação e avaliação.

5.1. Reconhecimento do software

A ferramenta foi apresentada pelo professor, em laboratório, e os alunos realizaram um reconhecimento inicial de sua interface. Esta etapa foi realizada para solucionar os problemas de incompreensão da mesma que pudessem ocorrer e influenciar a avaliação das funcionalidades do software. Cumpre salientar que não houve no experimento a intenção de avaliar a usabilidade da interface.

5.2. Experimentação

A seguir realizou-se a experimentação do software , onde os alunos foram incentivados a utilizar os três módulos do software na seguinte ordem: Tutorial, Experimentação Livre e Desafio. O professor buscou atuar como mediador principalmente quando os alunos estava realizando os desafios. O experimento teve a duração de noventa e cinco minutos.

5.3. Avaliação

Ao final da experimentação os alunos preencheram um instrumento de avaliação (questionário) sobre a ferramenta. A análise das respostas indicou que os alunos

encontraram facilidade para construção dos autômatos e para visualização da equivalência entre AFD e AFND, seja na forma de Tabela ou Diagrama de Transição. Todos assinalaram que a ferramenta auxiliou a desempenhar a tarefa.

6. Considerações Finais

O EduLing foi concebido para reduzir a dificuldade que muitos alunos apresentam para a aprendizagem das Linguagens Formais. Desta forma, o software busca melhorar o processo de aprendizagem nos cursos de Ciência da Computação.

O número de alunos que avaliaram a ferramenta ainda é pequeno para poder-se apresentar resultados conclusivos, porém alguns indicadores de que o software pode influenciar positivamente a aprendizagem de linguagens regulares foram encontrados.

Através do experimento foram identificados alguns pontos a serem melhorados. A linguagem do tutorial deve ser adaptada para a situação de ausência do professor possibilitando a auto-instrução, a representação gráfica dos autômatos deve permitir mais pontos de ligação entre os estados, deve ser construído um módulo de ajuda e desenvolvidos uma maior quantidade de exercícios no modulo tutorial.

Como trabalho futuro, pretende-se transformar o ambiente em um Sistema Tutor Inteligente capaz de realizar a avaliação automática das linguagens construídas pelo aluno no módulo desafio, adaptando os conteúdos e estratégias ao ritmo e perfil deste.

7. Referências Bibliográficas

AHO, Alfred V., et al.. (1995) “Compiladores: princípios, técnicas e ferramentas”, Tradução: Daniel de Ariosto Pinto, LTC – Livros Técnicos e Científicos.

CHOMSKY, Noam. (1955) “Logical Structure of Linguistic Theory”, MIT Humanities Library.

COHEN, Daniel. (1996) “Introduction to computer theory”, Wiley & Sons, Inc..

CRESPO, Rui Gustavo. (1998) “Processadores de linguagens: da concepção à implementação”, IST Press.

HOPCROFT, John E.; ULLMAN, Jeffrey D..(1979) “Introduction to automata theory, languages, and computation”, Addison-Wesley Publishing Company.

MENEZES, Paulo Blauth. (1997) “Linguagens formais e autômatos”, II – UFRGS, Sagra Luzzatto.

OLIVEIRA, Celina Couto de, et al.. (2001) “Ambientes informatizados de aprendizagem: produção e avaliação de software educativo”, Papirus, p. 144.

ROZENBERG, Grzegorz; SALOMAA, Arto. (1997) “Handbook of formal languages”, Springer-Verlag.