Projeto e Implementação de um Software para auxilio no Ensino de Linguagens Regulares

José Robson de Assis[1], Marlon Oliveira da Silva[1.]

Faculdade de Ciências Exatas e Comunicação (FACEC)

Universidade Presidente Antônio Carlos (UNIPAC) – Barbacena - MG

robsbq@hotmail.com, marlonos.br@gmail.com

Resumo. *Este artigo apresenta a modelagem e a implementação do SisLR – Sistema de Linguagens Regulares, software educativo voltado para o auxílio no ensino das linguagens regulares. O SisLR permite ao aluno a criação e validação dos autômatos através de uma interface gráfica interativa e de fácil usabilidade, tentando assim amenizar a dificuldade encontrada pelos alunos quando iniciam o estudo das linguagens formais.*

1. Introdução

A teoria das linguagens formais foi originalmente desenvolvida na década de 1950 com o objetivo de desenvolver teorias relacionadas com as linguagens naturais. Após esse estudo foi verificado que essa teoria apresentava fundamentos teóricos essenciais para o estudo das linguagens artificiais (linguagens regulares, livres de contexto), em especial, para as linguagens originárias da computação (MENEZES, 2005). Linguagens regulares, livres de contexto, sensíveis ao contexto e recursivamente enumeráveis são assuntos abordados pelo estudo das linguagens formais.

Linguagens formais é uma disciplina em que a maioria dos alunos apresentam dificuldades no início de seus estudos, isso devido ao alto grau de abstração exigido, principalmente no estudo das linguagens regulares no qual é introduzido o conceito de autômatos. Isso pôde ser comprovado através de uma pesquisa, detalhada na seção 5 desse trabalho, feita com alunos do 6º período do curso de Ciência da Computação da Universidade Presidente Antonio Carlos (UNIPAC) que estão estudando a disciplina.

Uma possível solução para tentar amenizar essa dificuldade encontrada pelos alunos, está no uso de softwares educativos que possibilitem a simulação, a criação e a validação de autômatos, através de uma interface gráfica de grande usabilidade. Este tipo de aplicação permite uma maior compreensão e aplicação dos conceitos estudados na teoria das linguagens formais.

Este trabalho apresenta a modelagem e a implementação de uma ferramenta educativa e interativa, denominada Sistema de Linguagens Regulares (SisLR), que auxilie no estudo de conteúdos relacionados a linguagens regulares.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica; a Seção 3 apresenta os trabalhos relacionados e suas características; a Seção 4 mostra a modelagem e a implementação da ferramenta SisLR; a Seção 5 apresenta análise e a discussão dos resultados obtidos com o uso da ferramenta. Finalmente, a Seção 6 conclui o artigo apresentando as sugestões de trabalhos futuros.

2. Fundamentação Teórica

Nesta seção são apresentados conceitos básicos relacionados ao estudo de linguagens regulares.

2.1. Linguagens Regulares

Linguagens regulares constituem a classe de linguagens com menor poder de representação, segundo a hierarquia de Chomsky, sendo possível desenvolver algoritmos de reconhecimento de grande eficiência e de fácil implementação. Há várias aplicações envolvendo o uso de Linguagens Regulares, como exemplo podemos citar a analise léxica, e mais recentemente sistemas de animação, hipertextos e hipermídias (MENEZES, 2005).

Linguagens regulares podem ser representadas por um Autômato Finito Determinístico (DFA - Deterministic Finite Automata), Autômato Finito Não Determinístico (NFA - Not Deterministic Finite Automata) e por uma Expressão Regular (ER).

2.2. Autômato Finito

Um autômato finito é um modelo matemático (máquina abstrata) de um sistema com entradas e saídas discretas. Pode ser determinístico (DFA) ou não-determinístico (NFA), no qual o não-determinístico significa que mais de uma transição para fora de um estado pode ser possível para o mesmo símbolo de entrada. Tanto os autômatos finitos determinísticos quanto os não determinísticos são capazes de reconhecer precisamente os conjuntos regulares (MENEZES, 2005).

2.2.1. Autômato Finito Determinístico (DFA)

Um autômato finito determinístico é uma estrutura matemática constituída de três tipos de entidades: um conjunto de estados, um alfabeto e um conjunto de transições. Dos estados, destaca-se um como o estado inicial, e destaca-se um subconjunto de estados como sendo composto dos estados finais. Um autômato finito Determinístico (DFA) é uma quíntupla (K,Σ,δ,q_0,F), no qual (VIEIRA,2004):

- K é um conjunto finito, não vazio, de estados;
- Σ: é um alfabeto finito de entrada;

- δ: K x $\Sigma \rightarrow$ K é a função de transição, função total;
- q_0: é denominado estado inicial e $q_0 \in K$;
- F é o conjunto de estados finais e F $\subseteq$ K.

2.2.2. Autômato Finito Não Determinístico (NFA)

Um NFA é uma abstração de um DFA, que permite a inserção do não-determinismo. Desta forma, pode haver múltiplas possibilidades de ação para um dado par estado X símbolo. Em um NFA, o não-determinismo pode ser implementado de duas formas: Múltiplas saídas com mesmo símbolo e transições em vazio (ξ) (MENEZES, 2005).

O processamento de um NFA é semelhante ao processamento de um DFA, apenas com uma diferenciação - um DFA processa em linha, enquanto um NFA processa através de uma árvore de possibilidades, em que se um dos ramos alcançarem um estado final, este será o caminho do processamento (MENEZES, 2005).

2.3 Expressões Regulares

Toda Linguagem regular pode ser descrita por uma expressão denominada expressão regular (ER). Trata-se de um formalismo denotacional, também considerado gerador, pois pode inferir como construir (gerar) as palavras de uma linguagem. Uma ER é definida recursivamente a partir de conjuntos (linguagens) básicos e operações de concatenação e união (MENEZES, 2005).

3. Trabalhos Relacionados

Nesta seção são apresentadas ferramentas de auxílio no ensino de linguagens regulares. Cada ferramenta será mostrada através de uma breve descrição, apontando suas principais características.

- **EduLing** (DOGNINI, 2003) - desenvolvida em Object Pascal, voltada para o auxílio e aprendizagem de linguagens regulares. O EduLing apresenta funções como: criação e validação de uma ER e de um DFA, transformação de ER para um DFA ou NFA e a de um DFA para uma ER. A ferramenta EduLing apresenta as seguintes características:
  - interface gráfica interativa para criação e manipulação de autômatos;
  - tutorial sobre a disciplina que auxilia o aluno na compreensão da mesma;
  - permite salvar e carregar os autômatos criados;
  - permite a visualização passo a passo quando da validação de um DFA;
  - apresenta limitações na parte de expressões regulares e autômatos;
  - não permite a criação e validação de um NFA.

- **Auger** (SLYMANSKI, 2004) - desenvolvida em Object Pascal, voltada para o apoio ao ensino de autômatos finitos. O Auger apresenta funções como: criação de um DFA ou NFA, validação de um DFA, transformação de um NFA para um DFA e a minimização de um DFA. A ferramenta Auger apresenta as seguintes características:
    - o software apresenta uma interface gráfica interativa para a criação e manipulação de autômatos;
    - possui uma ajuda de como utilizar a ferramenta explicando o funcionamento do software;
    - tutorial sobre a disciplina;
    - permite salvar e carregar os autômatos criados;
    - visualizar passo a passo ou continuamente a validação de um DFA;
    - não permite a manipulação de ER's e não permite a validação de NFA.
- **Language Emulador** (VIEIRA; VIEIRA; VIEIRA, 2002) - desenvolvida em Java, voltada para o auxilio no ensino de linguagens formais. Language Emulator apresenta como funções: criação e manipulação de ER's, gramáticas regulares, DFA's, NFA's, máquinas de Moore e de Mealy e suas equivalências. A ferramenta Language Emulador apresenta as seguintes características:
    - a ferramenta não apresenta uma interface gráfica interativa para criação e manipulação de autômatos;
    - não permite salvar e carregar os autômatos criados;
    - apresenta limitações na validação de um NFA;
- **Java Formal Language and Automata Package – JFLAP** (OLIVEIRA, 2005) - desenvolvida em Java, voltada para o auxílio no estudo Linguagens Formais. JFLAP é uma ferramenta que permite criar e simular diversos tipos de autômatos e converter diferentes representações de linguagens. A ferramenta JFLAP apresenta as seguintes características:
    - o software apresenta uma interface gráfica interativa para a criação e manipulação de autômatos;
    - uma ajuda de como utilizar a ferramenta;
    - permite salvar as linguagens criadas;
    - é uma excelente ferramenta para usuários já familiarizados com o assunto, mas torna-se confusa a quem esta iniciando por abranger várias linguagens formais.

4. Modelagem e implementação da ferramenta SisLR.

O Sistema de Linguagens Regulares, doravante denominado SisLR, foi desenvolvido para plataforma Windows usando o paradigma de programação orientada a objetos da linguagem Object Pascal do Delphi.

Para a fase de projeto do SisLR foi utilizada a Linguagem de Modelagem Unificada (UML - *Unified Modeling Language*) na qual os diagramas foram construídos utilizando-se a ferramenta *case* ARGOUML (ARGOUML, 2007).

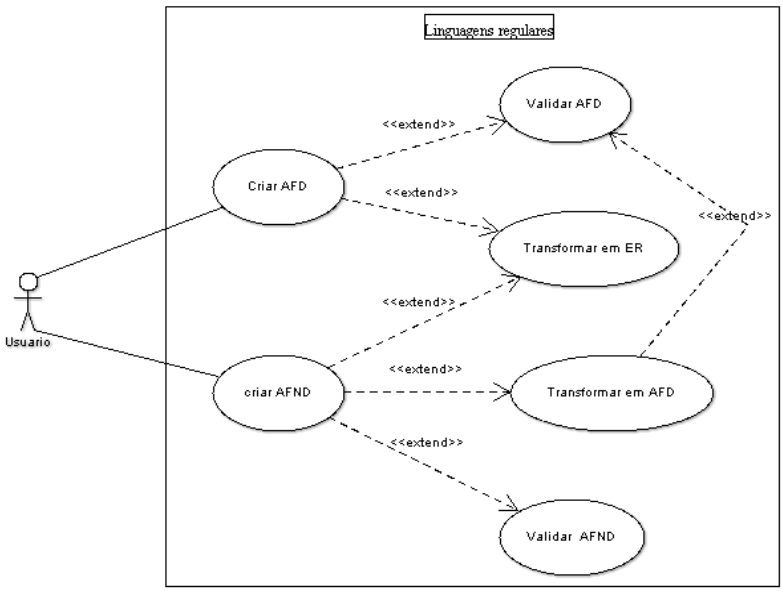

Figura 1. Diagrama de Casos de Uso do SisLR.

O SisLR foi concebido para ser uma aplicação que possibilitasse ao usuário manipular DFA's, NFA's e ER's e suas equivalências (transformações entre estes formalismos), como pode ser visto no diagrama de casos de uso, (Figura 1). Também disponibiliza ao usuário um tutorial sobre a disciplina que explica cada tópico da disciplina com exemplos e exercícios, além de uma ajuda de como utilizar a ferramenta.

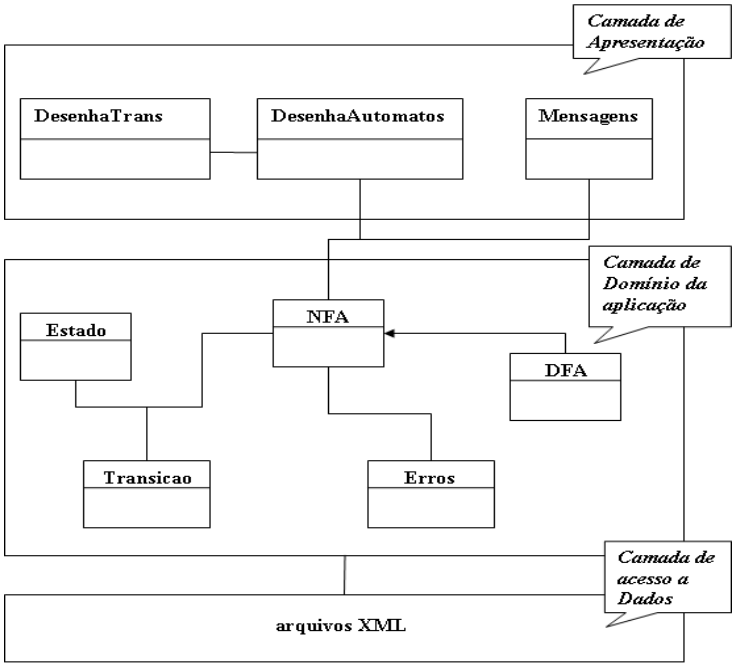

Figura 2. Arquitetura em três camadas.

O SisLR foi modelado como uma arquitetura em três camadas (Figura 2), o que permitiu uma modularização do sistema. Por exemplo, a interface gráfica de criação e manipulação de autômatos, representados na camada de apresentação, é independente, ou seja, permite que a interface de desenho dos autômatos possa ser modificada ou redesenhada sem a necessidade de alterar outras partes do software.

A camada de apresentação contém três classes:

1. *DesenhaTrans*: responsável por guardar as informações gráficas das transições (p. ex., coordenadas dos componentes no editor gráfico);
2. *DesenhaAutomato*: onde são implementados os algoritmos para criação e manipulação dos autômatos na tela;
3. *Mensagens*: cria e exibe as mensagens de ajuda ao usuário durante a execução do programa.

A camada de domínio da aplicação abrange toda a parte lógica do SisLR, onde são implementadas as principais funções do programa através das seguintes classes:

1. *Estado:* armazena informações referentes a um estado dentro de um autômato, como: nome e tipo;
2. *Transição*: faz referência a uma transição dentro de um autômato;
3. *Erros*: responsável por verificar e notificar erros cometidos na criação e manipulação de autômatos;

4. *NFA*: implementa todas as operações referentes a um NFA, como os algoritmos para: validar um NFA, transformá-lo em ER e transformá-lo em DFA;
5. *DFA:* classe filha da classe NFA, os métodos sobrescritos são usados para validar e transformar um DFA em ER.

A camada de acesso aos dados permite o armazenamento e a recuperação de arquivos XML que descrevem os autômatos criados pelo usuário. Arquivos XML possibilitam uma interoperabilidade maior entre ferramentas de criação e edição de autômatos. Este módulo encontra-se em desenvolvimento.

Figura 3. Tela inicial do SisLR.

O funcionamento do SisLR ocorre do seguinte modo, a ferramenta permite ao usuário determinar inicialmente se deseja criar um NFA ou DFA, (Figura 3) em seguida solicita informações do autômato, como linguagem e alfabeto (Figura 4).

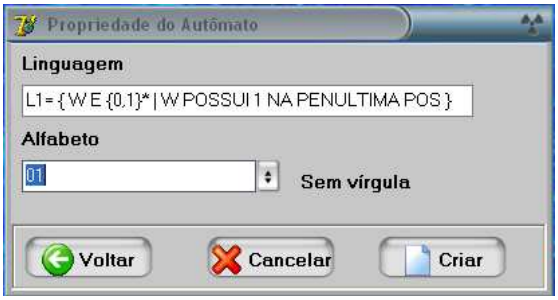

Figura 4. Propriedades do Autômato.

O SisLR permite criar autômatos através de uma interface gráfica de fácil manipulação, ou seja, a ferramenta possui um editor gráfico onde o usuário adiciona estados e faz transições com movimentos simples do mouse.

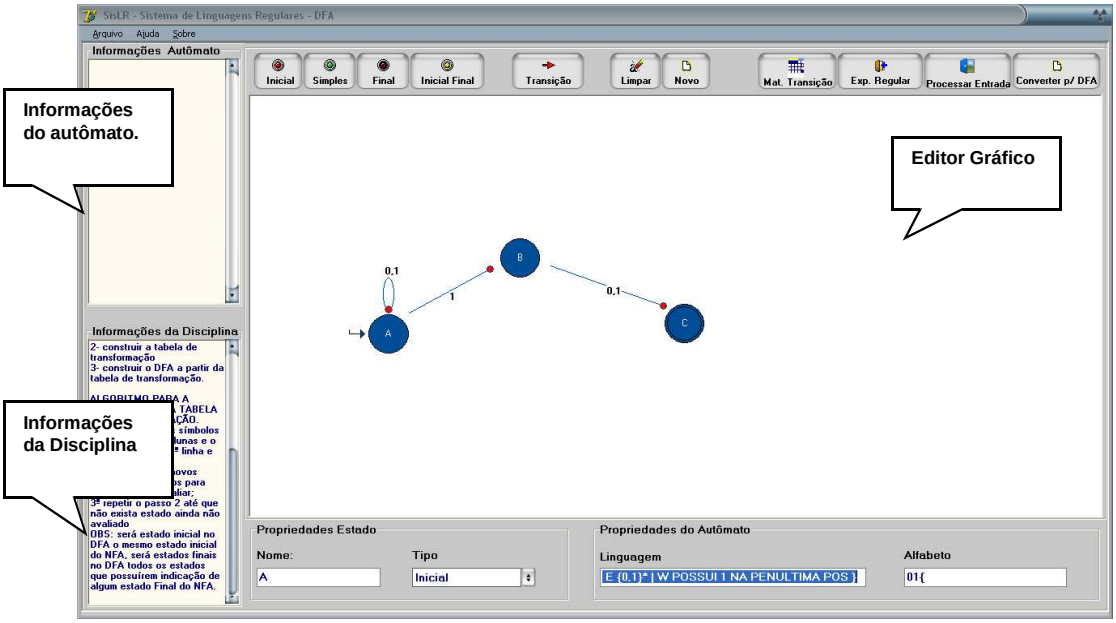

Figura 5. Tela principal do SisLR.

A Figura 5 mostra a tela principal do software, onde podemos visualizar o editor gráfico e outros componentes importantes do sistema, como: o campo informações do autômato que mostra os erros ocorridos quando da criação de um autômato e o campo informações da disciplina que exibe ao usuário informações relacionadas ao conteúdo de linguagens regulares.

Outra característica importante do SisLR é a forma como ele valida tanto um DFA quanto um NFA, a ferramenta permite ao usuário processar sua entrada de duas formas:

1. validação passo a passo: aonde o usuário determina o andar do processamento da entrada (Figura 6);

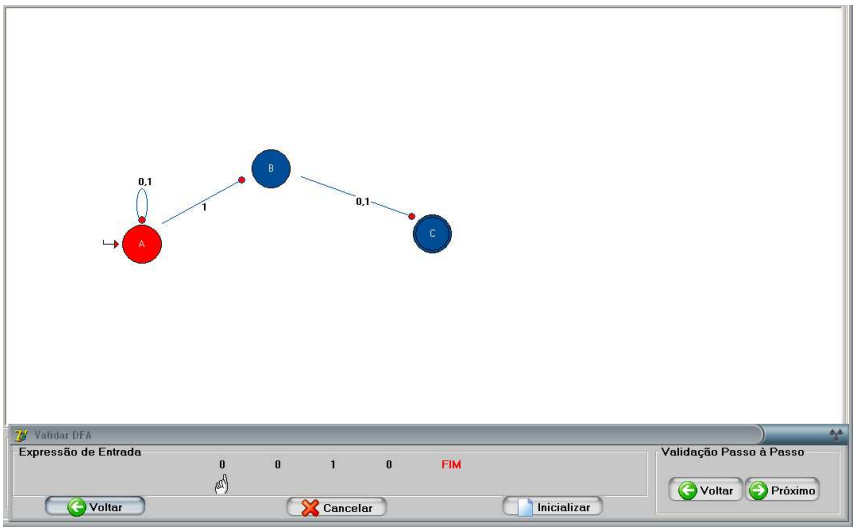

Figura 6. Validação Passo a Passo.

2. validação por tempo: aonde o usuário configura um tempo para que seja feita o processamento da entrada (Figura 7).

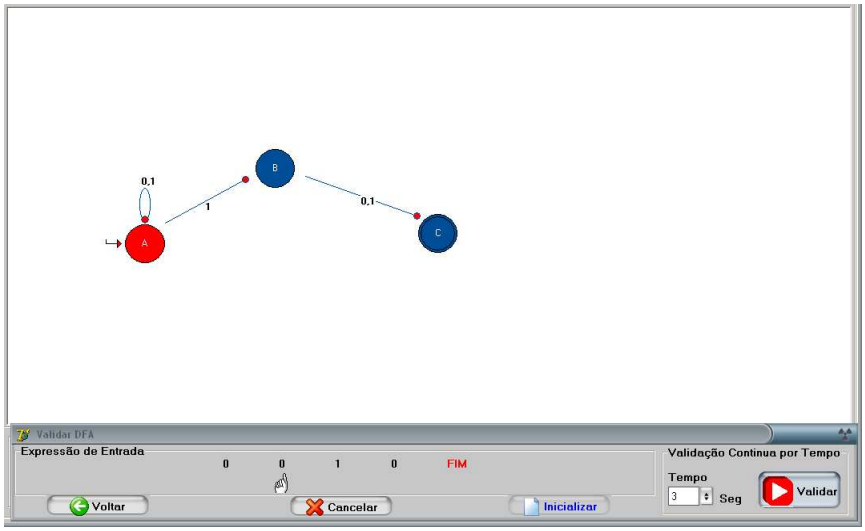

Figura 7. Validação por Tempo.

Com esse tipo de interface o usuário poderá visualizar melhor como é feito um processamento de um DFA ou NFA, verificando se sua entrada foi rejeitada ou não pelo seu autômato.

A ferramenta permite também: visualizar a matriz de transição do autômato, converter um DFA ou NFA em expressão regular e, no caso de se criar um NFA, permite a conversão graficamente de um NFA para um DFA.

O SisLR apresenta como diferencial em relação aos softwares descritos na Seção 3 a junção de características consideradas relevantes para um software de auxílio no ensino de linguagens regulares, como:

1. interface gráfica para criação e manipulação de autômatos: permite ao usuário criar seus autômatos de forma fácil e interativa;
2. operar com NFA, DFA e ER: aplicar as principais funções envolvendo esses formalismos, como: validar NFA ou DFA, transformar NFA em DFA e transformar um NFA ou DFA em ER;
3. tutorial da disciplina: material de ajuda abordando conceitos, exemplos e exercícios relacionados ao ensino de linguagens regulares;
4. ajuda do Sistema: manual explicando as funcionalidades do software;
5. salvar autômatos: permitir ao usuário carregar e salvar seus autômatos.

A Tabela 1 mostra um quadro comparativo entre os softwares da Seção 3 e o SisLR aonde são avaliadas as características apontadas anteriormente.

Tabela 1. Comparação entre os sistemas de linguagens regulares.

Características	EduLing	Auger	Language Emulador	JFLAP	SisLR
Interface gráfica para criação e manipulação de autômatos	Sim	Sim	Não	Sim	*Sim*
Operar com NFA, DFA e ER	Não	Não	Sim	Sim	*Sim*
Tutorial da Disciplina	Sim	Sim	Não	Não	*Sim*
Ajuda do Sistema	Sim	Sim	Não	Sim	*Sim*
Salvar Autômatos	Sim	Sim	Não	Sim	*Sim*

5. Análise e Discussão dos Resultados Obtidos

Com a finalidade de avaliar a funcionalidade e necessidade do software proposto, realizou-se uma pesquisa envolvendo 29 alunos do 6º período do curso Ciência da Computação da Universidade Presidente Antonio Carlos – UNIPAC. A pesquisa deu-se através das seguintes etapas.

1. Elaboração de um questionário composto por seis perguntas, que possibilitaram obter um percentual do nível de entendimento dos alunos em seis atividades relacionadas à compreensão e manipulação de autômatos finitos. Para cada pergunta foram dispostas cinco alternativas: 1. muito difícil; 2. difícil; 3. regular; 4. fácil; 5. muito fácil. Cada questão investigou o grau de cada atividade, a saber:
    a. criar um DFA;
    b. criar um NFA;
    c. transformar um NFA em DFA;
    d. visualizar o processamento de um DFA;
    e. visualizar o processamento de um NFA;
    f. diferenciar um DFA de um NFA.
2. Aplicação do questionário em sala de aula e análise dos resultados obtidos.
3. Apresentação do software para os alunos em laboratório mostrando suas funcionalidades através de exemplos e exercícios.
4. Aplicação do mesmo questionário em sala de aula e análise dos resultados obtidos, após o uso e conhecimento do SisLR.

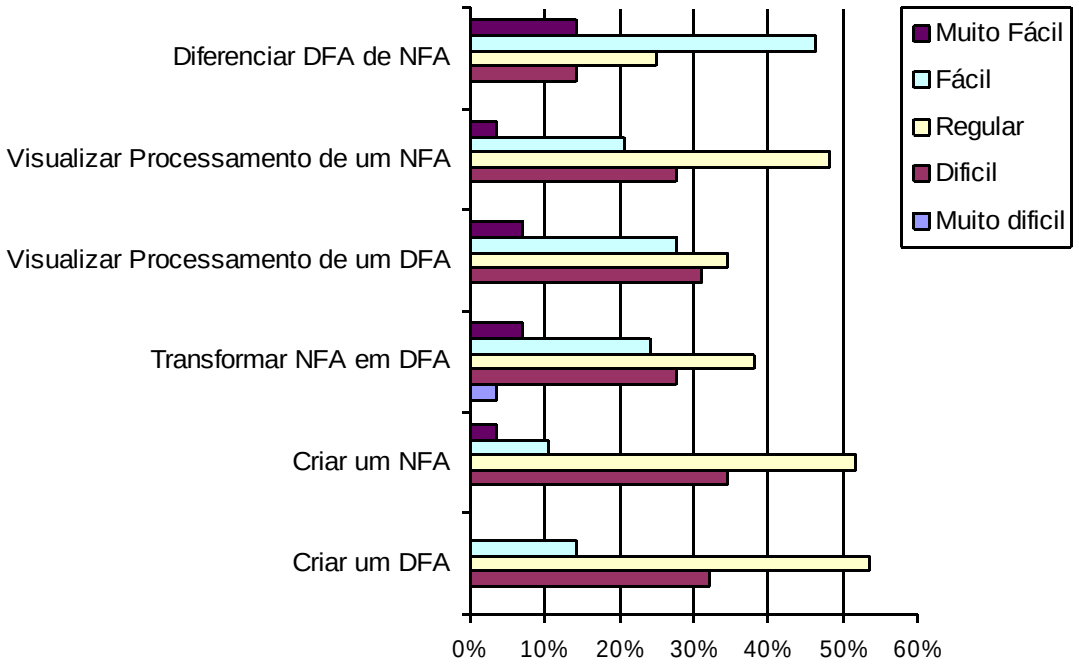

Figura 8. Gráfico gerado antes do uso SisLR.

A Figura 8 mostra o gráfico gerado na 2º etapa da pesquisa, podemos notar que em média 30 % dos alunos apresentaram dificuldade em realizar as cinco primeiras tarefas, e que somente a última foi considerada pela maioria, quase 50 %, uma tarefa fácil de realizar.

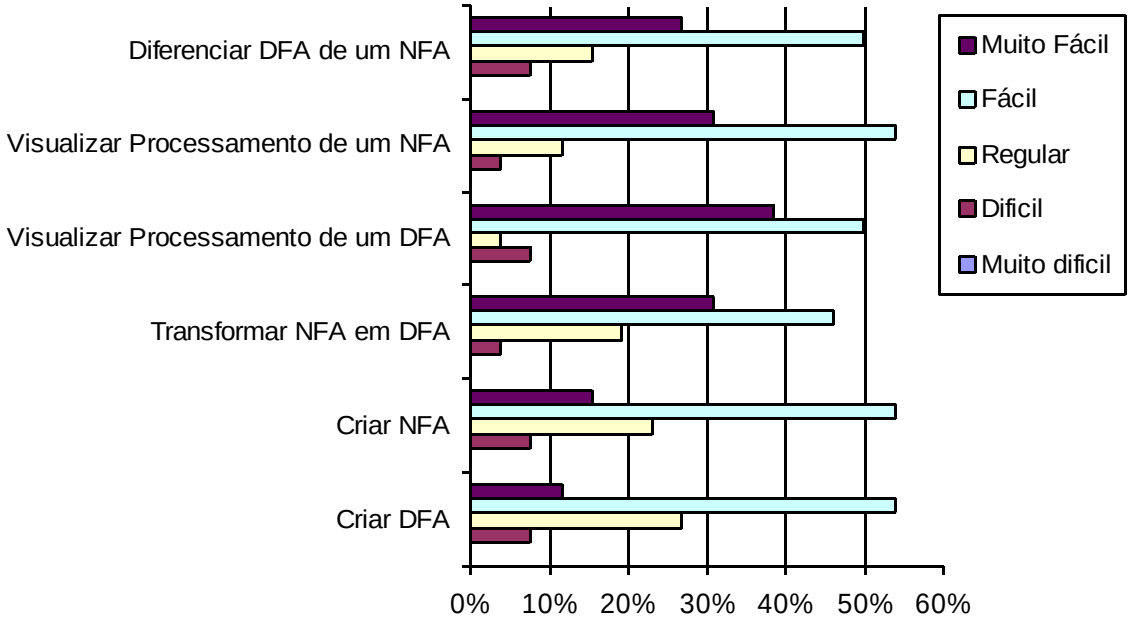

Figura 9. Gráfico gerado após uso do SisLR.

A Figura 9 mostra o gráfico gerado na última etapa da pesquisa, nota-se uma diminuição, em média de 20%, no nível de dificuldade encontrado pelos alunos nas

cinco primeiras atividades antes do uso do SisLR e um aumento considerável no nível de facilidade para realizar as mesmas.

A pesquisa mostrou que uso do SisLR aplicado junto ao estudo das linguagens regulares constitui uma ferramenta de grande ajuda ao aluno, pois contribui para o seu melhor desenvolvimento e aprendizado.

6. Conclusão e Trabalhos Futuros

Neste trabalho foi mostrado como o uso de ferramentas educativas no ensino de linguagens formais pode contribuir para o aprendizado do aluno, dando a ele a oportunidade de familiarizar-se mais com o assunto, podendo assim aplicar os conceitos envolvidos de forma simples e de fácil aprendizado.

Como proposta para desenvolvimento do trabalho foi desenvolvido o software Sistema de Linguagens Regulares (SisLR), para o auxílio no ensino de linguagens regulares.

As seguintes sugestões são feitas para trabalhos futuros no SisLR:

1. desenvolvimento da camada de acesso a dados usando arquivos XML para possibilitar a importação e exportação dos autômatos, o que favorece a interoperabilidade com outros sistemas de linguagens regulares;
2. aperfeiçoamento do mecanismo de ajuda do SisLR, através do uso de assistentes de *software (wizards)*, com o uso de técnicas de inteligência artificial que permitem a ferramenta sugerir ao usuário caminhos alternativos na construção do seu autômato;
3. implementação do algoritmo de minimização de um DFA, operação feita após a transformação de um NFA para DFA com intuito de encontrar um autômato equivalente com o menor numero de estados possíveis;
4. aperfeiçoamento da interface gráfica para manipulação de autômatos, através da implementação de técnicas modernas para visualização de grandes quantidades de dados (autômatos com muitos nós).
5. elaboração de exercícios aonde o software avalie o nível de aprendizado do aluno, juntamente com a emissão de um relatório mostrando a análise percentual dos erros e acertos do aluno.
6. desenvolvimento de um aplicativo web, possibilitando um maior alcance e conhecimento da ferramenta.

7. Bibliografia

ARGOUML (1996). *Software Livre. Versão 0.24. Tigris.* Fevereiro 2007. Disponível em: <http://argouml.tigris.org>. Acesso em: junho de 2007.

DOGNINI, José Marlon. (2003). “EduLing – Software Educacional para Linguagens Regulares”. Univali, Itajaí – SC.

MENEZES, Paulo Blauth. (2005). “Linguagens Formais e Autômatos”, 5º Edição, UFRGS, Sagra Luzzato.

OLIVEIRA, R. A. Rabelo (2005). “JFLAP JAVA Formal and Automata Package”. Disponível em http://homepages.dcc.ufmg.br/~nvieira/cursos/tl/a05s2/. Acessado em junho de 2007.

SZYMANSKI, Charbel. (2004). “Auger – Ferramenta de apoio ao ensino e autômatos finitos”. Unisul, Santa Catarina – SC.

VIEIRA, J. Newton. (2004). “Linguagens e Máquinas: Uma Introdução aos Fundamentos da Computação”. Departamento de Ciências Computação, UFMG.

VIEIRA, L. F. Menezes.; VIEIRA, M. A. Menezes.; VIEIRA, N. José. (2002). “Language Emulator, uma ferramenta de auxílio no ensino de Teoria da Computação”. UFMG.