Uma Ferramenta para Ensinar Introdução à Teoria da Computação

Candido Ferreira Xavier de Mendonça[1], **Luciano A. Digiampietri**[1]

[1]Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (USP)
Av. Arlindo Béttio, 1000 – 03828-000 – São Paulo – SP – Brasil

{cfxavier,digiampietri}@usp.br

Abstract. *Keeping students engaged from the beginning is one of the most important challenges in the undergraduate lectures. This challenge becomes more critical in theoretical subjects. This work reports the application of the concept of "proximal development zone" of the theory of Vygotsky, where a tool ("toy") is applied to mediate student learning conflicts in the "Introduction to the Theory of Computation" subject.*

Resumo. *Manter a motivação e o interesse do aluno desde o primeiro dia de aula é um dos maiores desafios da educação superior. O qual fica mais acentuado nas disciplinas teóricas. Este trabalho reporta a aplicação do conceito de "zona de desenvolvimento proximal" da teoria de Vygotsky na construção de uma ferramenta ("brinquedo") utilizada como proposta para mediar conflitos dos alunos na aprendizagem do conteúdo da disciplina "Introdução à Teoria da Computação".*

1. Introdução

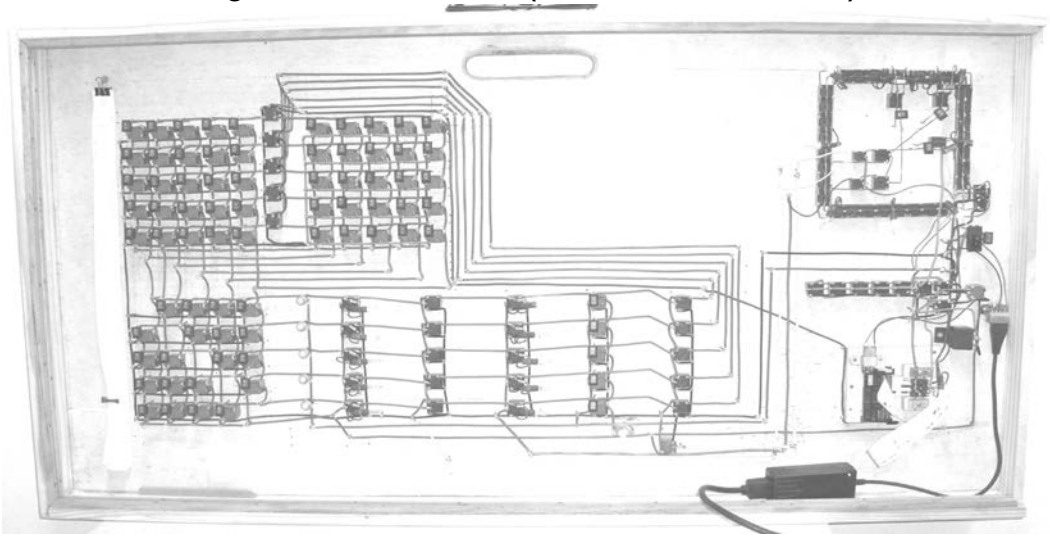
A disciplina Introdução à Teoria da Computação na maioria dos bacharelados da área de computação consiste no estudo da complexidade de três modelos computacionais teóricos: o autômato finito, o autômato à pilha e a máquina de Turing, com livros textos dentro dos quais destaca-se "Introdução à Teoria da Computação" [7].

Professores da disciplina "Introdução à Teoria da Computação" reclamam de um certo "desconforto" de alguns alunos na questão "modelos teóricos". Este desconforto fica mais evidente no início da apresentação de cada um dos 3 modelos computacionais da disciplina. Naturalmente, observamos que este comportamento não aparece em disciplinas práticas. Assim, o desafio é apresentar a disciplina buscando uma mediação para os conflitos citados.

Uma das propostas de remediação para esse desafio é dada por Vygotsky [2] onde o "brinquedo/brincadeira" (no caso, a ferramenta construída) é utilizado para melhorar o potencial de aprendizagem (definido por Vygotsky como "zona do desenvolvimento proximal"). Assim, para aplicar a Teoria de Vygotsky na disciplina Introdução à Teoria da Computação, seria necessário encontrar uma ferramenta adequada.

Existem excelentes ferramentas, como JFLAP [1, 6] desenvolvida por Susan Rodger, que podem ser utilizadas na disciplina em questão, porém, não são de caráter introdutório. Portanto, a escolha foi criar um material próprio, construindo um autômato

Figura 1. Foto do AFNDP (dimensões: 120 $\times$ 60 cm).



finito não-determinístico programável, doravante denominado por AFNDP[1], como pode ser visto na Figura 1. Este protótipo consiste numa máquina física, permitindo a implementação de qualquer autômato finito seja ele "determinístico", "não-determinístico" ou "não-determinístico com ε-transições", contendo até 5 estados, reconhecendo entradas do alfabeto $\{0,1\}$ registradas em uma fita de papel perfurado de 25mm.

Ao longo deste trabalho definiremos o conjunto do AFNDP mais sua programação como *autômato finito*, explicando assim não apenas o funcionamento do AFNDP, mas também a sequência de sua utilização ao longo das aulas.

O restante deste trabalho é organizado como segue: a seção 2 descreve a máquina e seu funcionamento, incluindo a tabela de execução para uma dada entrada; a seção 3 descreve a máquina e a tabela de execução para mostrar a equivalência entre o autômato finito "determinístico" e o autômato finito "não-determinístico"; a seção 4 descreve o uso da tabela de execução da máquina na demonstração do "Pumping Lemma" e a seção 5 conclui este trabalho.

2. A Ferramenta AFNDP

O AFNDP é apresentado logo na primeira aula, iniciando com um breve histórico dos três modelos computacionais que serão apresentados dentro da disciplina, isto é, o autômato finito, o autômato à pilha e a máquina de Turing. Este histórico é apresentado a seguir numa forma resumida.

Os dois primeiros modelos computacionais: autômato finito e autômato à pilha, foram criados por McCulloch e Pitts [4] em 1943, devido ao fato dos autores estarem estudando o comportamento do sistema nervoso, eles aparecem na literatura com o nome de neurônios. Treze anos mais tarde (1956), no trabalho de Kleene [3], esses neurônios serão renomeados para *Autômatos Finitos*, no qual é mostrada a relação entre Autômatos Finitos e Expressões Regulares, isto é, modelos com um tamanho limitado de memória que independe do tamanho da entrada. Três anos mais tarde (1959), Rabin e Scott [5]

[1]O diagrama esquemático do AFNDP está disponível em: 143.107.58.249/html/afnd-t2-r.eps

publicaram um estudo bastante completo sobre a Teoria dos Autômatos Finitos. Este último trabalho foi condecorado com "Turing Award", a maior premiação dada a trabalhos em Ciência da Computação. O terceiro modelo computacional, a máquina de Turing, foi criado por Alan Turing em 1936 e apresentado em seminários nos anos subsequentes [8, 9]. Curiosamente, a criação do modelo mais complexo antecede em 7 anos a criação dos dois modelos mais simples.

Este histórico é utilizado com o propósito de contextualizar o AFNDP. Dentro deste contexto foi criada uma máquina em que foram utilizados componentes eletromecânicos encontrados no final do século XIX, a saber: 1 motor bipolar, 151 relés, 20 lâmpadas, motores bipolares, 80 chaves, 11 campainhas, condutores e um mecanismo de impressão de uma máquina de cartão de crédito (transformado em leitora de fita de papel perfurado). Além disso, o projeto foi executado de modo semelhante a um trabalho de adolescente apresentado em uma feira de ciências do segundo grau.

O componente fundamental neste projeto é o relé de 5 terminais – um dispositivo eletromecânico responsável pela comutação de contatos elétricos, como mostrado na Figura 2. Seu funcionamento é bem simples: quando não há corrente entre os terminais da bobina, a mola de rearme projeta a armadura para "baixo", fazendo o contato entre os polos C e NF. Neste caso dizemos que o relé está *inativo*. Porém, na presença da corrente entre os terminais da bobina, um campo magnético é formado, atraindo a armadura, que se projeta para "cima", fazendo o contato entre os polos C e NA. Neste caso dizemos que o relé está *ativo*.

Figura 2. Um relé de 5 terminais.

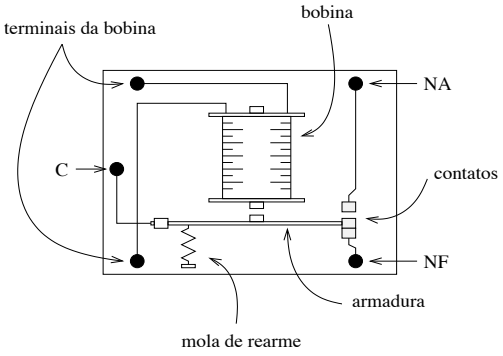
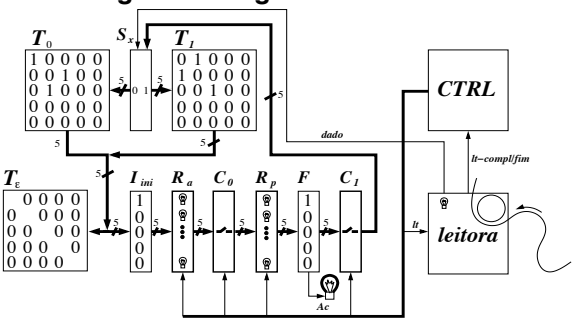


Figura 3. Diagrama de blocos.



2.1. O funcionamento do AFNDP

O diagrama de blocos funcionais do AFNDP pode ser visto na Figura 3. Os blocos funcionais do AFNDP são divididos em: tabelas, registradores e blocos de controle. As tabelas são caracterizadas pela presença de uma chave anexa a cada relé – exceto a tabela I_{ini} que consiste de uma coluna de campainhas. Os registradores são caracterizados pela presença de uma lâmpada anexa a cada relé do registrador, esta lâmpada serve para indicar se num dado momento, o relé correspondente está ativo ou não. Os blocos de controle não possuem uma característica definida, mesclando o uso de relés, lâmpadas e chaves de maneira não uniforme. Esses blocos funcionais estão dispostos em duas linhas: superior e inferior. A linha superior contém a tabela T_0, o seletor S_X, a tabela T_1 e a unidade de controle $CTRL$. A linha inferior contém a tabela T_ε, a tabela de ativação inicial I_{ini}, o registrador auxiliar R_a, o bloco de conexão C_0, o registrador principal R_p, a tabela F, a lâmpada Ac

para indicar estado de aceitação do AFNDP, o bloco de conexão C_1 e a leitora de papel perfurado.

A programação do autômato finito deve ser feita por meio das tabelas: I_{ini}, T_0, T_1, T_ε e F, onde um "1" significa que a chave (ou campainha) correspondente foi acionada, e um "0" significa que a chave ou campainha correspondente não foi acionada. A tabela I_{ini} é a única tabela com campainhas que devem ser acionadas somente no início da execução, indicando que o estado correspondente inicia "ativo". Um "1" na linha i e coluna j da tabela T_0 (resp. T_1 ou T_ε) indica a presença, durante a execução, da transição do estado q_i para o estado q_j para a entrada "0" (resp. transição para a entrada "1" ou transição-ε). Naturalmente, um "0" indica, que durante a execução, a transição não estará presente. A tabela F contém uma coluna com 5 chaves, onde um "1" na linha i significa, que durante a execução, a chave da linha i estará ligada (o estado q_i é um estado final). Neste caso, se o estado q_i estiver ativo, a lâmpada Ac acende indicando estado de "aceitação". Naturalmente, um "0" na linha i significa que a chave estará desligada (o estado q_i não é um estado final).

Assim, a programação exemplificada no AFNDP mostrada na Figura 3, a qual chamaremos de autômato finito M_1, pode ser descrita pelo diagrama de estados mostrado na Figura 4, onde usamos a forma pictórica de círculo para cada estado e arco para cada transição entre estados, estas transições dependem do caractere obtido na leitora. Inicialmente, acionamos a campainha referente ao estado q_0 (a tabela I_{ini} possui um "1" na linha 0), isso é indicado no diagrama por uma seta vindo do nada apontando para o estado q_0. Sempre que o símbolo "1" aparecer na linha i e coluna j na tabela T_0 (resp. tabela T_1 ou tabela T_ε) isso é indicado por um arco com rótulo "0" (resp. "1" ou "ε") ligando o estado q_i ao estado q_j no diagrama de estados. Finalmente, sempre que o símbolo "1" aparece na linha i da tabela F, isso indica que a lâmpada Ac ficará acesa caso este estado esteja ativo, isto é indicado no diagrama marcando o estado q_0 com duas circunferências concêntricas. Note que o diagrama de estados define claramente a programação do AFNDP.

Figura 4. Diagrama de estados do AFD M_1.

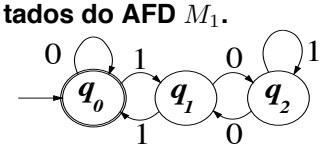
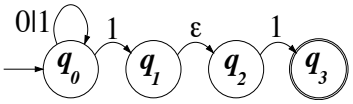


Figura 5. Diagrama de estados do AFND M_2.



A seguir, mostramos o funcionamento do AFNDP para um autômato finito qualquer, onde a tabela T_ε contém apenas zeros, e, como não desempenha nenhuma função, será omitida.

Inicialização:

Quando o AFNDP é ligado a conexão C_0 está ativada, os dois registradores R_a e R_p estão habilitados e vazios, as tabelas: T_0, T_1 e F estão desabilitadas e a unidade $leitora$ está posicionada no início da fita de papel. Primeiramente, o estado inicial escolhido é ativado usando uma das 5 campainhas da tabela I_{ini}, que será registrado em R_a e R_p. Na sequência, ativamos a unidade de controle $CTRL$ para que execute ciclicamente as transições.

Execução de uma transição:

A unidade de controle $CTRL$ inicia cada transição com a conexão C_0 ativada, isto é,

o registrador auxiliar R_a terá um conteúdo idêntico ao do registrador principal R_p. Na sequência, a conexão C_0 é desativada e a conexão C_1 é ativada[2]. Simultaneamente, um sinal de controle lt é passado para a unidade $leitora$ para que obtenha o próximo caractere na fita de papel. Durante a leitura, as tabelas: T_0 e T_1 são desabilitadas e o registrador auxiliar R_a é apagado. Quando a unidade $leitora$ completa o ciclo de leitura do próximo caractere ($dado$), este é enviado para o seletor S_X e um sinal de completude do ciclo de leitura $lt - compl$ é enviado para a unidade de controle $CTRL$, que irá habilitar as tabelas: T_0 e T_1 e o registrador auxiliar R_a. Na sequência, o seletor S_X, fazendo uso do $dado$ enviado pela unidade $leitora$, seleciona a tabela que deve receber o conteúdo do registrador principal R_p. Independentemente de qual seja a tabela usada, a saída é registrada no registrador auxiliar R_a. Na sequência, a unidade de controle $CTRL$ apaga o conteúdo do registrador principal R_p, ativa a conexão C_0 e habilita o registrador principal R_p, que irá copiar o conteúdo do registrador auxiliar R_a.

Final da execução:

No caso em que a fita de papel termine durante a execução do ciclo de leitura, no meio da execução de uma transição, um sinal fim é enviado à unidade de controle $CTRL$, que interrompe o processamento. Neste momento, caso a lâmpada Ac esteja acesa, o AFNDP termina em estado de "aceitação", caso contrário (se a lâmpada Ac estiver apagada) o AFNDP termina em estado de "rejeição".

A título de exemplo, vamos acompanhar a execução do autômato M_1 para a palavra "1001", que pode ser visto na Tabela 1. Inicialmente, pressionamos a campainha da linha 0 de I_{ini}, ativando a unidade de controle $CTRL$ para que execute as transições. O primeiro caractere lido é um "1". Logo, S_X irá colocar o conteúdo de R_p na entrada da tabela T_1, onde q_0 ativa q_1. No final desta transição, o estado q_1 é o único estado ativo. O próximo caractere lido é um "0". Logo, S_X irá colocar o conteúdo de R_p na entrada da tabela T_0, onde q_1 ativa q_2. O próximo caractere lido é um "0". Logo, S_X irá colocar o conteúdo de R_p na entrada da tabela T_0, onde q_2 ativa q_1. O próximo caractere lido é um "1". Logo, S_X irá colocar o conteúdo de R_p na entrada da tabela T_1, onde q_1 ativa q_0. Por fim, como a cadeia de entrada terminou, o AFNDP finaliza a execução. Entretanto, como o estado q_0 está ativo e este estado é final (a chave 0 da tabela F está ligada), a lâmpada Ac está acesa, indicando que a cadeia "1001" foi aceita. De fato, essa cadeia é a representação binária do número 9 e o autômato implementado no exemplo aceita uma representação binária de um múltiplo de 3, incluindo a cadeia vazia.

Tabela 1. Exec. para a entrada "1001"

	I_{ini}	1	0	0	1
q_0	x				x
q_1		x		x	
q_2			x		

Tabela 2. Definição de ψ para M_1

$\psi =$

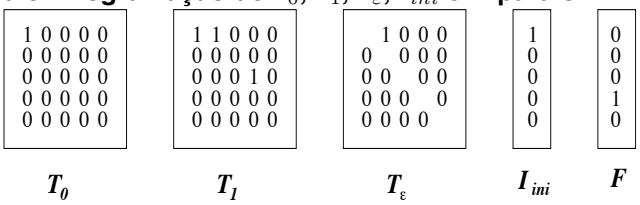
ψ	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

O autômato M_1 pode ser definido formalmente como: $M_1 = (\{q_0, q_1, q_2\}, \{0, 1\}, q_0, \{q_0\}, \psi)$, onde ψ é definida pela Tabela 2.

Observamos que o AFNDP aqui mostrado somente permite a implementações de autômatos de no máximo 5 estados. Porém, caso o AFNDP não possuísse tal limitação, perguntamos: Que tipo de linguagens o autômato finito reconhece? Que tipo de linguagens o autômato finito não reconhece?

[2]as duas conexões C_0 e C_1 são complementares, quando uma é ativada a outra é desativada e vice-versa

Figura 6. Programação de T_0, T_1, T_ε, I_{ini} e F para o AFND M_2.



3. AFD e AFND são equivalentes

Embora a máquina apresentada tenha o nome de AFNDP (Autômato Finito Não-Determinístico Programável). O modelo de programação e funcionamento mostrados, como veremos a seguir, é "determinístico". Porém, como desejamos responder a pergunta "Que tipo de linguagens o AFNDP reconhece?", precisamos explorar todas as possibilidades. Para isso devemos fazer as seguintes definições.

Dizemos que um autômato finito é *determinístico* se existe uma única instância do AFNDP em execução, onde um e somente um estado está ativo no final de cada transição.

Dizemos que um autômato finito é *não-determinístico* se durante a execução do AFNDP é possível haver mais de uma instância em funcionamento sincronizado na mesma posição para a mesma cadeia de entrada onde cada instância possui estados ativos q_i e q_j diferentes. Porém, é fácil demonstrar que as múltiplas instâncias citadas são equivalentes a ter uma única instância do autômato finito com ambos os estados q_i e q_j ativos.

Um exemplo de diagrama de estados de um autômato não-determinístico (autômato M_2) pode ser visto Figura 5 e a programação de sua implementação no AFNDP pode ser vista na Figura 6.

O autômato M_2 pode ser definido formalmente como: $M_2 = (\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \{q_0\}, \{q_3\}, \psi)$, onde ψ é definida pela Tabela 3.

Tabela 3. Def. de ψ para M_2

$\psi =$

ψ	0	1	ε
q_0	$\{q_0\}$	$\{q_0,q_1\}$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\emptyset$
q_2	$\emptyset$	$\{q_3\}$	$\emptyset$
q_3	$\emptyset$	$\emptyset$	$\emptyset$

Tabela 4. Execução de M_3 para "1011"

	I_{ini}	E	1	E	0	E	1	E	1	E
q_0	x	x	x	x	x	x	x	x	x	x
q_1			x	x			x	x	x	x
q_2				x				x		x
q_3									x	x

Durante a execução do AFNDP para um autômato não-determinístico, caso este contenha transições-ε, haverá uma mudança na execução de cada transição, pois a tabela T_ε pode acrescentar novos estados ativos ao registrador auxiliar R_a antes deste ser copiado no registrador principal R_p. Deste modo, o diagrama de execução passa a ter uma nova coluna E para registrar estes acréscimos. Um exemplo do diagrama de execução do autômato finito não determinístico M_2 para a cadeia de entrada "1011" pode ser visto na Tabela 4. Note que toda vez que aparece um caractere "1" no estado q_1, o estado q_2 é ativado na coluna E correspondente.

A execução da Tabela 4 (o AFNDP para M_2) pode ser explicada como segue. Inicialmente ativamos o estado q_0, em seguida, copiamos este estado ativo na coluna E. Como não existe nenhuma transição-ε saindo de nenhum estado ativo para um estado

inativo, então, esta coluna E é uma cópia da coluna anterior. Em seguida, o caractere "1" é lido, para este caractere o estado q_0 ativa os estados q_0 e q_1. Estes dois estados são copiados na próxima coluna E. Como existe uma transição-ε de q_1 (ativo) para q_2 (inativo), este estado q_2 é marcado como ativo. Em seguida o caractere "0" é lido, para este caractere, o estado q_0 é o único estado a ser ativado. Esta coluna é copiada na coluna E seguinte. Novamente, não existe nenhuma transição-ε saindo de nenhum estado ativo para um estado inativo, logo esta coluna E é uma cópia da coluna anterior. Em seguida, o caractere "1" é lido, para este caractere e o estado q_0 ativa os estados q_0 e q_1. Estes dois estados são copiados na próxima coluna E. Como existe uma transição-ε saindo de q_1 (ativo) para q_2 (inativo), este estado q_2 é marcado como ativo. Em seguida, o caractere "1" é lido, para este caractere o estado q_0 ativa os estados q_0 e q_1 e o estado q_2 ativa o estado q_3. Estes três estados são copiados na próxima coluna E. Como existe uma transição-ε saindo de q_1 (ativo) para q_2 (inativo), este estado q_2 é marcado como ativo. Neste momento o AFNDP termina o processamento em estado de aceitação, pois o estado final q_3 está ativo e a lâmpada Ac está acesa.

Agora, vamos construir a Tabela 5 que consiste no conjunto de estados ativos durante a execução do AFNDP, onde a linha Q_i indica os estados ativos no final de cada transição i, para o autômato finito não-determinístico M_2, com uma entrada qualquer.

Iniciamos a Tabela 5, linha dois, coluna um, com o conjunto que pode ser obtido por meio de uma nova tabela (Tabela 6). Essa tabela é construída da seguinte forma. Na primeira coluna colocamos todos os estados do autônomo M_2 e na última linha da segunda coluna, incluímos o conjunto unitário contendo o estado inicial de M_2. Em seguida marcamos como ativo esse estado. Essa coluna é copiada na coluna seguinte (E). Como não existe nenhuma transição-ε (em M_2) de q_0 para um estado inativo, a coluna E é uma cópia da coluna I_{ini}. Por fim, criamos um conjunto na última linha da coluna E contendo todos os estados marcados nesta coluna (no exemplo, $\{q_0\}$). Esse valor é transportado para a Tabela 5. Feito isto, é necessário completar as duas últimas colunas desta segunda linha. O procedimento para o preenchimento destas colunas é feito como segue. Cria-se uma tabela contendo quatro colunas. Na primeira coluna colocamos todos os estados do autônomo M_2 e na última linha da segunda coluna, incluímos o conjunto indicado na primeira coluna da Tabela 5. Marcamos nesta coluna como ativos todos os estados que aparecem nesse conjunto. Colocamos na primeira linha, segunda coluna, o caractere "0" (resp. "1") que seria lido na cadeia de entrada. Nessa coluna é indicado que estados seriam ativados por M_2 para o caractere "0" (resp. "1"). A terceira coluna é copiada na coluna quatro (E) e verifica-se possíveis transições$-\varepsilon$ completando o conjunto de estados ativos. Todos os estados marcados nesta coluna são copiados no conjunto representado na última linha. Este conjunto é copiado na coluna indicada por "0" (resp. "1") da Tabela 5.

Tabela 5. Tabela de estados ativos do AFND M_2.

$\{Q_i\}$	0	1
$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1, q_2\}$
$\{q_0, q_1, q_2\}$	$\{q_0\}$	$\{q_0, q_1, q_2, q_3\}$
$\{q_0, q_1, q_2, q_3\}$	$\{q_0\}$	$\{q_0, q_1, q_2, q_3\}$

Tabela 6. Estados ativos, no início e no final da transição-ε.

	I_{ini}	E
q_0	x	x
q_1		
q_2		
q_3		
	$\{q_0\}$	$\{q_0\}$

Usando esse procedimento, completamos a linha dois da Tabela 5, obtendo os conjuntos $\{q_0\}$ e $\{q_0, q_1, q_2\}$ conforme Tabela 7. Note que o novo conjunto $\{q_0, q_1, q_2\}$ foi transportado para a coluna um linha três da Tabela 5.

Tabela 7. Simulação do comportamento do AFNDP para as entradas "0" e "1"

		0	E
q_0	x	x	x
q_1			
q_2			
q_3			
	$\{q_0\}$		$\{q_0\}$

		1	E
q_0	x	x	x
q_1		x	x
q_2			x
q_3			
	$\{q_0\}$		$\{q_0, q_1, q_2\}$

Usando o mesmo procedimento, completamos a linha três da Tabela 5, obtendo os conjuntos $\{q_0\}$ e $\{q_0, q_1, q_2, q_3\}$ conforme Tabela 8. Note que o novo conjunto $\{q_0, q_1, q_2, q_3\}$ foi transportado para a coluna um linha quatro da Tabela 5.

Tabela 8. Simulação do comportamento do AFNDP para as entradas "0" e "1"

		0	E
q_0	x	x	x
q_1	x		
q_2	x		
q_3			
	$\{q_0, q_1, q_2\}$		$\{q_0\}$

		1	E
q_0	x	x	x
q_1	x	x	x
q_2	x		x
q_3		x	x
	$\{q_0, q_1, q_2\}$		$\{q_0, q_1, q_2, q_3\}$

Usando o mesmo procedimento, completamos a linha quatro da Tabela 5, obtendo os conjuntos $\{q_0\}$ e $\{q_0, q_1, q_2, q_3\}$ conforme Tabela 8. Note que, nenhum novo subconjunto do conjunto de estados Q do autômato M_2 aparece nas colunas dois e três da Tabela 5, e, portanto ela está completa[3].

Tabela 9. Simulação da execução do AFNDP para o autômato M_2 - estados ativos

		0	E
q_0	x	x	x
q_1	x		
q_2	x		
q_3			
	$\{q_0, q_1, q_2, q_3\}$		$\{q_0\}$

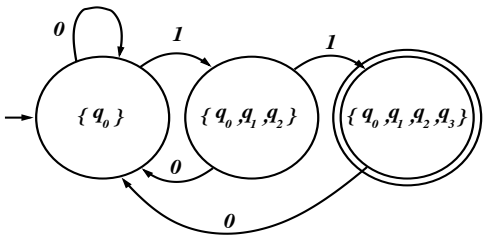
		1	E
q_0	x	x	x
q_1	x	x	x
q_2	x		x
q_3		x	x
	$\{q_0, q_1, q_2, q_3\}$		$\{q_0, q_1, q_2, q_3\}$

O comportamento dos estados ativos no AFNDP na simulação da execução do autômato finito não-determinístico M_2, para uma entrada genérica, pode ser exemplificado pelo diagrama de estados ativos mostrado na Figura 7. Porém, este diagrama de estados é o diagrama de um autômato finito determinístico. Não é difícil mostrar que isto é verdade para qualquer AFND, logo o AFD e o AFND são equivalentes.

Embora o não-determinismo não acrescente nenhum poder novo ao AFNDP, ele nos ajuda a responder a primeira pergunta. Neste ponto, seguindo os melhores livros de Linguagens Formais, introduzindo as Expressões Regulares e mostrando a sua equivalência com autômatos não-determinísticos. Com isto, respondemos "Que tipo de linguagens o AFNDP reconhece?".

[3]Observamos que, no pior caso, o número de conjuntos que pode aparecer nessa tabela é $2^{|Q|}$, isto é, no máximo o número de subconjuntos do conjunto Q.

Figura 7. Diagrama de estados ativos durante a execução do AFNDP para autômato M_2.



4. Pumping Lemma

O "Pumping Lemma" é peça chave para responder: "O que não é uma linguagem regular?" (não existe um autômato que reconheça a linguagem).

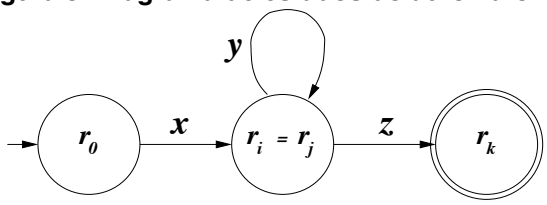
Lema 1. *Se L é uma linguagem regular então existe um número n tal que toda palavra s pertencente à linguagem L, onde $|s| \geq n$, pode ser dividida em três sub-palavras x, y e z, tal que:*
(i) $|xy| \leq n$, (ii) $|y| > 0$ e (iii) $xy^*z \in L$.

Demonstração. Seja L uma linguagem regular e $M' = (Q, \Sigma, q_0, F, \psi)$ um autômato finito determinístico que reconhece L. Seja n o número de estados do autômato M', isto é, $n = |Q|$. Seja $s = s_1s_2s_3...s_k$ uma palavra qualquer de L tal que $|s| = k \geq n$. Agora, executamos o AFNDP para os primeiros n caracteres de s, conforme a Tabela 10. Note que renomeamos os estados no final de cada transição (última linha da tabela), renomeando para r_0 o estado ativo inicial (coluna I_{ini}), para r_i o estado ativo na transição onde o caractere s_i foi lido. Então, contamos o número de estados que ficaram ativos durante esse processo. Essa contagem dá $n + 1$, pois temos $n + 1$, contando as colunas de 0 à n. Porém, existem apenas n estados no autômato M', logo, pelo menos um destes estados é repetido durante essa execução. Sejam i o menor valor tal que o estado r_i se repete durante a execução. Seja r_j a primeira repetição de r_i. Sejam x a sub-palavra formada pelos caracteres $s_1s_2...s_i$, y a sub-palavra formada pelos caracteres $s_{i+1}s_{i+2}...s_j$ e y a sub-palavra formada pelos caracteres $s_{j+1}s_{j+2}...s_k$. Naturalmente, pela escolha de i e j as sub-palavras x e y satisfazem (i) e (ii), pois existe uma repetição de estado garantida dentro das primeiras n transições. Ora, a palavra s pertence à linguagem, logo, o estado r_k pertence à F. Além disso r_j é uma repetição de r_i, logo, quando a execução chega na transição i (no final da sub-palavra x) podemos repetir qualquer quantidade de vezes a palavra y (inclusive zero vezes) sempre terminando com o estado $r_i = r_j$ ativo. Assim, se executarmos M' para a sub-palavra z chegaremos no final com r_k ativo (veja Figura 8). Logo, $xy^*z \in L$ o que satisfaz (iii). □

Tabela 10. Execução do AFNDP para os primeiros n caracteres de s.

	I_{ini}	s_1	s_2	s_3	...	s_n	s_{n+1}	...	s_k
q_0	x				...			...	
q_1		x			...			...	
q_2			x		...			...	
...	...	...	...	...	...	x		...	
q_{n-1}				x	...			...	
	r_0	r_1	r_2	r_3	...	r_n	r_{n+1}	...	r_k

Figura 8. Diagrama de estados do autômato M'.



5. Conclusão

Por se tratar de um protótipo, a máquina foi apenas apresentada aos alunos. A apresentação do AFNDP na primeira aula[4] de Introdução à Teoria da Computação tem se mostrado uma excelente estratégia de ensino nos últimos oito anos. Não poucas vezes, após o término da aula às 22:30h, alguns alunos se detinham, curiosos sobre a máquina.

O diagrama de execução aqui descrito facilita a introdução tanto da forma determinística como da forma não-deterministica dos autômatos finitos e ajuda na demostração de sua equivalência. Também, simplifica a demonstração do "Pumping Lemma".

A experiência têm mostrado que a disciplina iniciada com o AFNDP consegue manter um número bastante elevado de alunos concentrados até o último dia de aula.

Referências

[1] D. Caugherty and S. H. Rodger. NPDA: A tool for visualizing and simulating nondeterministic pushdown automata. In *DIMACS Workshop*, pages 365–377, March 1992.

[2] M. K. de Oliveira. *Pensamento e Ação no Magistério: Vygotsky, Aprendizado e desenvolvimento, Um processo sócio-histórico*, chapter 4, pages 55–79. Editora Scipione, São Paulo, 2006.

[3] S. C. Kleene. Representation of events in nerve nets and finite automata. *Automata Studies*, 1956.

[4] W. S. McCulloch and W. H. P. Jr. A logical calculus of ideas immanent in nervous activity. *Bull. Math. BioPhys.*, 5:115–133, 1943.

[5] M. O. Rabin and D. Scott. Finite automata and their decision problems. *IBM Journal of Research and Development archive*, 3 Issue 2:114–125, 1959.

[6] S. H. Rodger, E. Wiebe, K. M. Lee, C. Morgan, K. Omar, and J. Su. Increasing engagement in automata theory with jflap. In *SIGCSE '09 Proceedings of the 40th ACM technical symposium on Computer science education*, pages 403–407. ACM New York, NY, USA, March 2009.

[7] M. Sipser. *Introdução à Teoria da Computação*. Thompson Pioneira, Brasil, 2007.

[8] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. In *Proc. London Math. Soc. Ser. 2 42*, pages 230–265, 1937.

[9] A. M. Turing. Correction to: On computable numbers, with an application to the entscheidungsproblem. In *Proc. London Math. Soc. Ser. 2 43*, pages 544–546, 1938.

[4] O conteúdo da Seção 2 é coberto na primeira aula (aprox. 1:45h).