3.5 Exercícios

3.1 Apresente a representação na forma de tabela de transições para o autômato finito da Figura 3.4.

3.2 Construa um autômato finito determinístico com um número mínimo de estados para reconhecer sentenças descritas pela expressão $a(ab)*b$. Utilize os procedimentos formais para obter o autômato finito não-determinístico, convertê-lo para um autômato finito determinístico e minimizar seu número de estados.

3.3 Dada a expressão regular $(xx)*(y|z)z*$

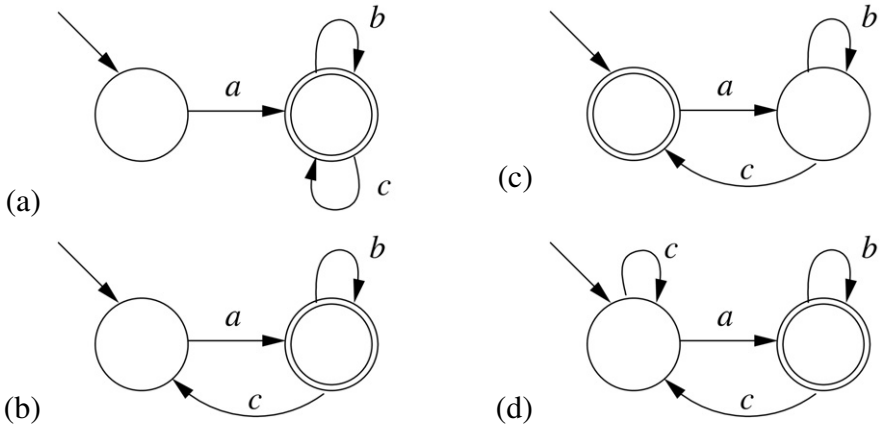
(a) Construa, usando o algoritmo de Thompson, o autômato finito não-determinístico para reconhecer sentenças dessa linguagem.

(b) Converta, usando o método da construção de subconjuntos, o autômato do item *a* para um autômato finito determinístico.

(c) Minimize, se possível, o número de estados do autômato do item *b*.

3.4 Desenvolva o autômato finito determinístico com o menor número de estados para reconhecer sentenças da gramática regular $G = \{V_n, V_t, P, A\}$, com símbolos não-terminais $V_n = \{A, B, C\}$, símbolos terminais $V_t = \{x, y\}$ e produções $P = \{A \rightarrow xB, A \rightarrow yB, B \rightarrow xC, C \rightarrow xC, C \rightarrow y\}$.

3.5 Apresente as expressões regulares que representam a linguagem reconhecida pelos seguintes autômatos finitos:



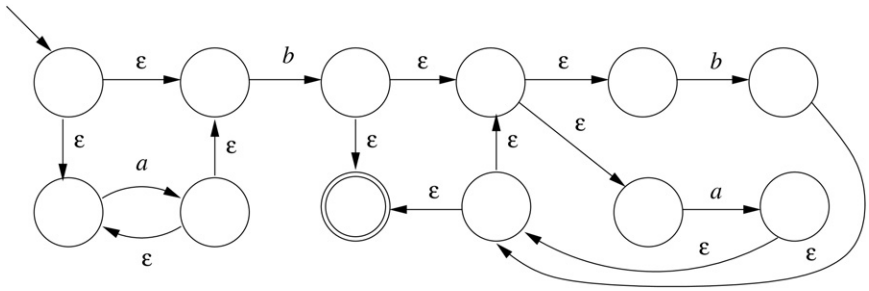
3.6 Em uma aplicação que aceita strings binárias (alfabeto: {0,1}), as sentenças aceitas são aquelas que terminam com o mesmo bit que iniciaram. Assim, se a expressão inicia com 0, pode ter qualquer combinação de bits (até mesmo vazia) após esse bit desde que haja outro bit 0 no final. Similarmente, sentenças iniciadas com 1 devem terminar com o bit 1.

(a) Apresente uma expressão regular que descreva essas sentenças.

(b) Desenvolva o autômato finito não-determinístico, usando o algoritmo de Thompson, para reconhecer essas sentenças.

(c) Converta o autômato obtido no item anterior para um autômato determinístico com número mínimo de estados.

3.7 A aplicação do algoritmo de Thompson produziu um autômato finito não-determinístico com a seguinte estrutura:



(a) Utilize o método da construção de subconjuntos e o procedimento de minimização de estados para obter um autômato finito determinístico equivalente.

(b) Qual é a expressão regular que descreve as sentenças reconhecidas por esses autômatos?

3.8 Usando a notação de metacaracteres de lex, descreva as expressões regulares que representam as seguintes sentenças:

(a) Todas as seqüências compostas por símbolos 0 e 1 tal que a ocorrência de um 0 é seguida por um ou mais 1's. Observe que a seqüência vazia também é válida.

(b) Uma seqüência que representa um número real, que pode ter um sinal, dígitos na parte inteira, parte fracionária (uma vírgula seguida por dígitos) e expoente (a letra 'e' ou 'E' seguida por um sinal opcional e um ou mais dígitos). Contemple na representação as possíveis combinações usuais para valores reais aceitas em linguagens de programação.

3.9 O seguinte arquivo em formato lex especifica tokens que serão aceitos como entrada em uma aplicação:

```
%%
0[01]{2}0
1(0{2}|1{2})[01]
```

Dada essa especificação, indique quais das seguintes entradas seriam rejeitadas ou aceitas pela aplicação — nesse caso, aponte se pela primeira ou pela segunda regra.

(a) 0000
(b) 0120
(c) 0001
(d) 0010
(e) 0210
(f) 1000
(g) 1010
(h) 1020
(i) 1111
(j) 100110

3.10 A biblioteca padrão de C contém as seguintes funções que realizam a conversão de strings que contêm a representação de valores numéricos para variáveis:

strtol(str, ret, base) Retorna o valor inteiro (tipo `long`) resultante da conversão da seqüência de caracteres iniciada em `char *str`, que deve ser iniciada com brancos ou valores numéricos válidos, de acordo com a `base` indicada. A primeira posição não reconhecida na string é retornada em `char **ret`, que pode ser 0 se esse valor não for utilizado. Caso a base seja 16, a string pode iniciar com o prefixo `0x`.

strtod(str, ret) Retorna o valor real (tipo `double`) resultante da conversão da seqüência de caracteres iniciada em `char *str`, que deve ser iniciada com brancos ou uma representação em ponto flutuante válida. A primeira posição não reconhecida na string é retornada em `char **ret`, que pode ser 0 se esse valor não for utilizado.

(a) Utilize essas funções para estender a aplicação apresentada na Seção 3.4.4 de forma que o total de todos os valores reconhecidos seja também calculado e apresentado pela aplicação.

(b) Modifique as expressões regulares na especificação do analisador léxico para que valores negativos também sejam reconhecidos e computados corretamente no total calculado pela aplicação.

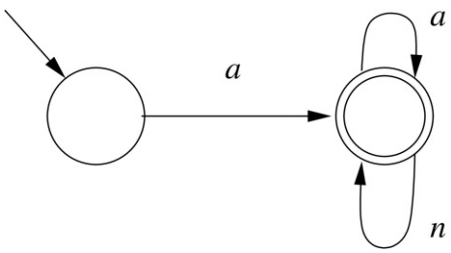
3.11 Apresente uma única expressão regular (a mais compacta possível) que seja capaz de descrever, com o auxílio de metacaracteres da linguagem lex:

(a) Qualquer palavra de 10 caracteres alfabéticos que inicie e termine por vogal;

(b) Qualquer nome de variável de até 5 caracteres alfanuméricos iniciado obrigatoriamente por caractere alfabético;

(c) Qualquer número inteiro, positivo ou negativo, em octal (iniciado por 0), hexadecimal (iniciado por 0x ou 0X) ou decimal (não iniciado por 0). Considere que não há representação para o inteiro zero em decimal.

3.12 Seja o seguinte autômato finito determinístico, capaz de interpretar expressões para uma gramática regular G:



(a) Qual a expressão regular que esse autômato determinístico reconhece?

(b) Faça a tabela de transições de estados para esse autômato.

(c) Escreva seis seqüências de símbolos reconhecíveis pelo autômato.

(d) Com base na expressão regular que você obteve, desenhe o autômato finito não-determinístico correspondente, mostrando as etapas intermediárias em sua construção.