

---

**Observações:**

- **O trabalho é individual.**
- Cada questão vale 1,5 pontos.
- A interpretação de cada questão faz parte do trabalho. Caso alguma questão esteja ambígua ou incorreta, ou caso falte(m) elemento(s) importante(s) para o seu entendimento, faça as suposições e/ou correções que achar convenientes e **explícite-as**.
- Na correção de cada questão será analisado se a mesma foi respondida com **rigor, clareza e objetividade**. O aluno perderá pontos nas seguintes situações, por exemplo, *mesmo se sua resposta estiver correta*:
  - usa frases incoerentes, ou sem sentido, ou incorretas, ou que nada tenham a ver com o que foi perguntado;
  - apresenta uma solução sem apresentar a(s) idéia(s) subjacente(s), dificultando a correção;
  - apresenta uma solução conceitualmente deslegante;
  - é prolixo, escrevendo coisas desnecessárias ao entendimento da solução.

Dependendo do “grau” de tais impropriedades, o aluno pode até perder todos os pontos relativos à questão.

- Devolução: até as 16:00 horas do dia 24/11/2003, na sala 4023, ou em meu escaninho na Recepção do DCC, ou via *email*.

---

1. Segue uma definição recursiva do conjunto dos palíndromos sobre  $\Sigma$ ,  $Pal$ :

- (a)  $\lambda \in Pal$ ;
- (b)  $a \in Pal$  para todo  $a \in \Sigma$ ;
- (c) se  $x \in Pal$  e  $a \in \Sigma$  então  $axa \in Pal$ ;
- (d) nenhuma palavra é membro de  $Pal$ , a menos que possa ser obtida como mostrado em a, b e c.

O conjunto dos palíndromos sobre  $\Sigma$  pode também ser definido utilizando-se a operação *reverso*:  $Pal' = \{w \in \Sigma^* \mid w = w^R\}$ . A operação *reverso*, por sua vez pode ser assim definida:

- (a)  $\lambda^R = \lambda$ ;
- (b)  $(xa)^R = ax^R$  para quaisquer  $x \in \Sigma^*$  e  $a \in \Sigma$ .

Prove que  $Pal = Pal'$ .

2. Seja, para  $n \geq 1$  e  $b \geq 2$ ,

$$L_{n,b} = \{x \in \{0, 1, \dots, b-1\}^* \mid \eta_b(x) \bmod n = 0\},$$

onde  $\eta_b(x)$  é o número representado por  $x$  na base  $b$ . Na Seção 2.1.2 da apostila foi mostrado um AFD para  $L_{6,2}$ . Em anexo, segue um trecho de Kozen<sup>1</sup> em que é mostrado como construir um AFD para  $L_{3,2}$ , e onde está desenvolvida uma prova da correção de tal AFD. Generalize o método e a prova para  $n$  e  $b$  arbitrários.

3. Seja a linguagem:

$$L = \{a^i b^j c^k \mid i \neq j \text{ ou } j \neq k\}.$$

Suponha que na aplicação do lema do bombeamento (LB) para provar que  $L$  não é regular, seja escolhida  $z = a^k b^{k!+k} c^{k!+k}$ , onde  $k$  é a constante do LB. Descubra a razão “misteriosa” para tal escolha.

4. Mostre que as linguagens regulares são fechadas sob a operação *metade*:

$$metade(L) = \{x \mid xy \in L \text{ e } |x| = |y|\}.$$

*Sugestão*: mostre como obter um AF  $M'$  tal que  $L(M') = metade(L)$ , a partir de um AF  $M$  para  $L$ ; em seguida, prove que  $M'$  reconhece  $metade(L)$ .

5. Construa AFN $\lambda$ 's para as linguagens a seguir, usando transições  $\lambda$  para *simplificar* a concepção.

- (a) O conjunto das palavras que consistem de zero ou mais **a**'s, seguidos por zero ou mais **b**'s, seguidos por zero ou mais **c**'s.
- (b) O conjunto das palavras que consistem, ou de 01 repetida uma ou mais vezes, ou de 010 repetida uma ou mais vezes.
- (c) O conjunto das palavras de 0's e 1's tais que pelo menos uma das dez últimas posições contém 1.

---

<sup>1</sup>Kozen, D.C. *Automata and Computability*, Springer, 1997.

## Lecture 4

### More on Regular Sets

Here is another example of a regular set that is a little harder than the example given last time. Consider the set

$$\{x \in \{0,1\}^* \mid x \text{ represents a multiple of three in binary}\} \quad (4.1)$$

(leading zeros permitted,  $\epsilon$  represents the number 0). For example, the following binary strings represent multiples of three and should be accepted:

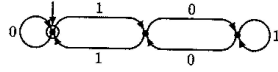
<i>Binary</i>	<i>Decimal equivalent</i>
0	0
11	3
110	6
1001	9
1100	12
1111	15
10010	18
$\vdots$	$\vdots$

Strings not representing multiples of three should be rejected. Here is an automaton accepting the set (4.1):

		0	1
→	0F	0	1
	1	2	0
	2	1	2

Figura 1: page 19

The states **0**, **1**, **2** are written in boldface to distinguish them from the input symbols 0, 1.



In the diagram, the states are **0**, **1**, **2** from left to right. We prove that this automaton accepts exactly the set (4.1) by induction on the length of the input string. First we associate a meaning to each state:

<i>if the number represented by the string scanned so far is<sup>1</sup></i>	<i>then the machine will be in state</i>
$0 \bmod 3$	<b>0</b>
$1 \bmod 3$	<b>1</b>
$2 \bmod 3$	<b>2</b>

Let  $\#x$  denote the number represented by string  $x$  in binary. For example,

$\#\epsilon = 0$ ,  
 $\#0 = 0$ ,  
 $\#11 = 3$ ,  
 $\#100 = 4$ ,

and so on. Formally, we want to show that for any string  $x$  in  $\{0, 1\}^*$ ,

$$\begin{aligned} \hat{\delta}(0, x) = 0 & \text{ iff } \#x \equiv 0 \bmod 3, \\ \hat{\delta}(0, x) = 1 & \text{ iff } \#x \equiv 1 \bmod 3, \\ \hat{\delta}(0, x) = 2 & \text{ iff } \#x \equiv 2 \bmod 3, \end{aligned} \quad (4.2)$$

or in short,

$$\hat{\delta}(0, x) = \#x \bmod 3. \quad (4.3)$$

This will be our induction hypothesis. The final result we want, namely (4.2), is a weaker consequence of (4.3), but we need the more general statement (4.3) for the induction hypothesis.

We have by elementary number theory that

$$\#(x0) = 2(\#x) + 0,$$

<sup>1</sup>Here  $a \bmod n$  denotes the remainder when dividing  $a$  by  $n$  using ordinary integer division. We also write  $a \equiv b \bmod n$  (read:  $a$  is congruent to  $b$  modulo  $n$ ) to mean that  $a$  and  $b$  have the same remainder when divided by  $n$ ; in other words, that  $n$  divides  $b - a$ . Note that  $a \equiv b \bmod n$  should be parsed  $(a \equiv b) \bmod n$ , and that in general  $a \equiv b \bmod n$  and  $a = b \bmod n$  mean different things. For example,  $7 \equiv 2 \bmod 5$  but not  $7 = 2 \bmod 5$ .

$$\#(x1) = 2(\#x) + 1,$$

or in short,

$$\#(xc) = 2(\#x) + c \quad (4.4)$$

for  $c \in \{0, 1\}$ . From the machine above, we see that for any state  $q \in \{0, 1, 2\}$  and input symbol  $c \in \{0, 1\}$ ,

$$\delta(q, c) = (2q + c) \bmod 3. \quad (4.5)$$

This can be verified by checking all six cases corresponding to possible choices of  $q$  and  $c$ . (In fact, (4.5) would have been a great way to *define* the transition function formally—then we wouldn't have had to prove it!) Now we use the inductive definition of  $\hat{\delta}$  to show (4.3) by induction on  $|x|$ .

*Basis*

For  $x = \epsilon$ ,

$$\begin{aligned} \hat{\delta}(0, \epsilon) &= 0 && \text{by definition of } \hat{\delta} \\ &= \# \epsilon && \text{since } \# \epsilon = 0 \\ &= \# \epsilon \bmod 3. \end{aligned}$$

*Induction step*

Assuming that (4.3) is true for  $x \in \{0, 1\}^*$ , we show that it is true for  $xc$ , where  $c \in \{0, 1\}$ .

$$\begin{aligned} \hat{\delta}(0, xc) &= \delta(\hat{\delta}(0, x), c) && \text{definition of } \hat{\delta} \\ &= \delta(\#x \bmod 3, c) && \text{induction hypothesis} \\ &= (2(\#x \bmod 3) + c) \bmod 3 && \text{by (4.5)} \\ &= (2(\#x) + c) \bmod 3 && \text{elementary number theory} \\ &= \#xc \bmod 3 && \text{by (4.4).} \end{aligned}$$

Note that each step has its reason. We used the definition of  $\delta$ , which is specific to this automaton; the definition of  $\hat{\delta}$  from  $\delta$ , which is the same for all automata; and elementary properties of numbers and strings.

#### Some Closure Properties of Regular Sets

For  $A, B \subseteq \Sigma^*$ , recall the following definitions:

$$\begin{aligned} A \cup B &= \{x \mid x \in A \text{ or } x \in B\} && \text{union} \\ A \cap B &= \{x \mid x \in A \text{ and } x \in B\} && \text{intersection} \\ \sim A &= \{x \in \Sigma^* \mid x \notin A\} && \text{complement} \end{aligned}$$