**SAC de Teoria da Computação — 2017/2**
**Prof. José de Oliveira Guimarães — DComp — Campus de Sorocaba da UFSCar**

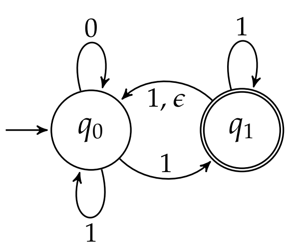
Ao final da prova, entregue APENAS a folha de respostas. A folha de questões não será considerada. Utilize quantas folhas de reposta forem necessárias. Não é necessário entregar as questões em ordem. Utilize lápis ou caneta, mas faça letra legível. Na correção, símbolos ou palavras ilegíveis não serão considerados.

**Justifique** todas as respostas.

Se você estudou com algum colega para esta prova, não se sente ao lado dele pois é possível que acidentalmente vocês produzam respostas semelhantes para alguma questão. Provas de alunos que fizeram a prova próximos uns dos outros com respostas semelhantes caracterizam cópia de questões. Lembro-os de que todos os envolvidos na utilização de métodos ilegais na realização desta prova receberão zero de nota final da disciplina (e não apenas nesta prova).

Coloque o seu nome na folha de resposta, o mais acima possível na folha, seguido do número da sua coluna de carteiras. A primeira carteira é a mais perto da porta e a última a mais perto das janelas. Não precisa colocar o RA.

1. (1,5) Explique o funcionamento de um autômato finito não determinístico (AFND) utilizando o exemplo abaixo. Nesta explicação, obviamente, deve ser mostrada a diferença para um autômato finito determinístico (AFD).




2. (3,5) Nesta questão serão utilizadas as linguagens $L_1 = \{ 0^n 1 \mid n \in \mathbb{N} \}$, $L_2 = \{ a^n 0 b^n \mid n \in \mathbb{N} \}$, $L_3 = \{ a^n w b^n \mid n \in \mathbb{N} \text{ e } w \in L_1 \}$. Responda às questões abaixo sobre gramáticas, expressões regulares e linguagens.

(a) (1,5) Faça gramáticas $G_1$, $G_2$ e $G_3$ tal que $L(G_1) = L_1$, $L(G_2) = L_2$ e $L(G_3) = L_3$.

(b) (1,0) Explique, informalmente mas precisamente, por que um autômato finito não pode reconhecer $L_2$.

(c) (1,0) Faça uma expressão regular (e.r.) que reconheça $L_1$ e uma outra que reconheça $L_1^\star$.

3. (1,0) Cite a Tese de Church-Turing.

4. (1,5) Uma MT de uma única fita possui apenas a instrução

$$(q_0, 0, q_s, 1, P)$$

Faça a codificação desta única instrução. Assuma que o alfabeto é $\Sigma = \{0, 1, \rhd, \sqcup, \square\}$. Não é necessário seguir as regras dadas na apostila. Mas lembre-se que uma codificação é uma função injetora e, dado um número $n \in \mathbb{N}$, deve ser possível descobrir se $n$ corresponde a uma MT e qual MT é esta. Não é preciso justificar formalmente, mas deixe claro o raciocínio que você seguiu (os passos para chegar na resposta).

Escolha e faça UMA e apenas UMA das questões abaixo.

5. (2,5) Prove que, se $L$ e $K$ são c.e., então $L$ é computável.

6. (2,5) Seja $K$ uma linguagem decidida por uma MT $M_k$ e $L$ uma linguagem decidida por uma MT $M_L$. $L \subset K$ e $L \neq K$. Prove que $K - L$ é uma linguagem decidível construíndo uma MT $M$ que a decide.

Resumo:

Uma expressão regular sobre um alfabeto $\Sigma$ é descrita como: a) $x$ é uma e.r. (expressão regular) se $x \in \Sigma$; b) $\epsilon$ é uma e.r. c) $\emptyset$ é uma e.r. d) $x?$, $(x \cup y)$, $(xy)$ (concatenação de $x$ e $y$) e $(x^\star)$ são e.r. se $x, y$ são e.r. Assume-se que a ordem de precedência dos operadores seja, do maior para o menor: $\star$, concatenação e união ($\cup$). Parenteses podem ser removidos se forem redundantes. A linguagem associada a uma expressão regular é definida indutivamente como $L(x) = \{x\}$ se $x \in \Sigma$, $L(\epsilon) = \{\epsilon\}$, $L(\emptyset) = \emptyset$, $L(R_1 \cup R_2) = L(R_1) \cup L(R_2)$, $L(R_1 \circ R_2) = L(R_1) \circ L(R_2)$, $L(R_1^\star) = (L(R_1))^\star$, $L(R?) = L(R) \cup \{\epsilon\}$.

A concatenação de duas linguagens $L$ e $K$ é $L \circ K = \{vw : v \in L \text{ e } w \in K\}$.

A definição de uma gramática pode ser feita com a seguinte sintaxe:

$A \longrightarrow 0A$
$A \longrightarrow BC$
$B \longrightarrow bB|\epsilon$
$C \longrightarrow c$

Uma máquina de Turing determinística (MT ou MTD) é uma quadrupla ($Q$, $\Sigma$, $I$, $q$) na qual $Q$ e $\Sigma$ são conjuntos chamados de conjuntos de estados e de símbolos, $I$ é um conjunto de instruções, $I \subset Q \times \Sigma \times Q \times \Sigma \times D$, $D = \{-1, 0, 1\}$ e $q \in Q$ é chamado de estado inicial. Há dois estados especiais: $q_s$ e $q_n$, todos elementos de $Q$ e todos diferentes entre si. Neste texto convenciona-se que o estado inicial será $q_0$ a menos de menção em contrário. Exige-se que $\{0, 1, \rhd, \sqcup, \square\} \subset \Sigma$. Uma instrução é da forma $(q_i, s_j, q_l, s_k, d)$ na qual $s_k \neq \square$ e $q_i \notin \{q_s, q_n\}$. Se $(q, s, q_0', s_0', d_0), (q, s, q_1', s_1', d_1) \in I$, então $q_0' = q_1'$, $s_0' = s_1'$ e $d_0 = d_1$. $Q$, $\Sigma$ e $I$ são conjuntos finitos.

O símbolo $\square$ é o branco utilizado para as células ainda não utilizadas durante a execução e $\sqcup$ é utilizado para separar dados de entrada e saída. A saída de uma MT é o conjunto de símbolos da célula corrente até o $\rhd$ que está à direita (deve existir alguma célula à direita da célula corrente com este símbolo).

Símbolos: Máquina de Turing Não Determinística (MTND), Máquina de Turing (MT), Máquina de Turing Determinística (MTD). A menos de menção em contrário, uma MT é uma MTD. Todas as linguagens utilizadas são subconjuntos de $\{0, 1\}^\star$.

Uma linguagem $L$ é computável (recursiva) se existe uma MT $M$ de decisão tal que

$$x \in L \text{ sse } M(x) = 1$$

Isto é, $M$ decide $L$.

Uma linguagem $L$ é ou computacionalmente enumerável, c.e. (recursivamente enumerável, r.e.) se existe uma MT $M$ tal que se $x \in L$ então $M(x) = 1$. Se $x \notin L$, $M(x) \uparrow$. Dizemos que $M$ *aceita* $L$.

Uma linguagem $L$ pertence a TIME($f$) se existe uma MT $M$ de decisão que toma uma entrada $x$ e que termina a sua execução depois de um número de passos menor ou igual a $cf(n)$ tal que $x \in L$ sse $M(x) = 1$. Ou seja, $M$ executa em tempo $cf(n)$. Uma linguagem $L$ pertence a SPACE($f$) se existe uma MT $M$ que toma uma entrada $x$ e que termina a sua execução depois de utilizar um número de células menor ou igual a $cf(n)$ nas fitas de trabalho (todas exceto a de entrada e a de saída). Ou seja, $M$ executa em espaço $cf(n)$. NTIME($f$) e NSPACE($f$) têm definições análogas, mas que usam MTND´s.

$$\mathrm{P} = \bigcup_{k \in \mathbb{N}} \mathrm{TIME}(n^k)$$