Prova Substitutiva de Teoria da Computação — 2018/2
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.

A menos de menção em contrário, usaremos o alfabeto $\Sigma = \{0, 1\}$ nesta prova.

Primeira Parte da Matéria

1. (2,0) Sobre gramáticas, responda.

(a) Uma gramática G tem as produções

$S \longrightarrow aSb \mid A$

$A \longrightarrow A0 \mid 1.$

Faça um Autômato com Pilha M tal que $L(M) = L(G)$. Obrigatoriamente construa o autômato de acordo com as regras do Sipser (com somente três estados).

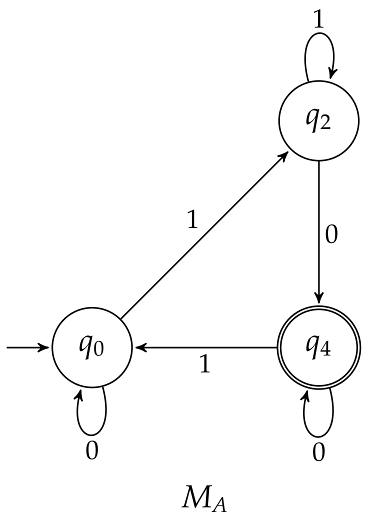
(b) Explique, sem uma prova formal, porque um autômato finito (sem pilha) não pode reconhecer $L(G)$.

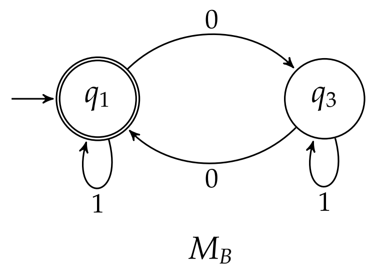
2. (2,0) Escolha e faça UM e APENAS UM dos itens abaixo.

(a) Seja M um autômato finito. Mostre como obter um autômato que reconheça $L(M) \cdot \{0\}^\star$. Faça apenas o desenho como está no Sipser, com retângulos com três círculos representando autômatos. Lembre-se de que, se A e B são linguagens, $A \cdot B = \{ab : a \in A \text{ e } b \in B\}$. E $\{0\}$ é uma linguagem com um único elemento.

(b) Usando o algoritmo dado em aula e no livro do Sipser, faça o Autômato Finito *Determinístico* que reconhece $L(M_A) \cup L(M_B)$, sendo M_A e M_B os autômatos dos diagramas dados abaixo.

Este algoritmo usa, para cada combinação de estados q_a de M_A e q_b de M_B, um estado (q_a, q_b).





3. (2,0) Escolha e faça UM e APENAS UM dos itens abaixo.

(a) Utilizando o Lema do bombeamento, prove que $L = \{0^n 1^n \mid n \in \mathbb{N}\}$ não é linguagem regular.

(b) Sendo $M_A = (Q_A, \Sigma_A, \delta_A, q_A, F_A)$ um Autômato Finito Determinístico, dê o domínio e contra-domínio de δ_A (algo assim: $\delta_A : X \longrightarrow Y$, explicitando X e Y). Sendo $M_B = (Q_B, \Sigma_B, \delta_B, q_B, F_B)$ um Autômato Finito Não-Determinístico, dê o domínio e contra-domínio de δ_B. Sendo $M_C = (Q_C, \Sigma_C, \Gamma_C, \delta_C, q_C, F_C)$ um Autômato com Pilha, dê o domínio e contra-domínio de δ_C.

(c) Defina linguagem de uma gramática $G = (V, \Sigma, P, S)$. Explique, usando um exemplo, porque gramáticas são não determinísticas.

Segunda Parte da Matéria

4. (2,0) Escolha e faça UM e APENAS UM dos itens abaixo.

(a) Prove que, se L e K são linguagens decidíveis, $L \cap K$ é *enumerável*.

(b) Prove que, se L e K são linguagens decidíveis, $L \cup K^c$ é decidível.

(c) Prove que, se L e K são linguagens decidíveis, $L - K^c$ é decidível.

5. (2,0) Sobre complexidade:

(a) (1,0) explique porque $TIME(f) \subset SPACE(f)$. Dê detalhes, muitos detalhes, preferencialmente um exemplo;

(b) (1,0) explique porque $SPACE(n) \subset SPACE(n^2)$.

6. (2,0) A MT P toma dois inteiros como parâmetros. O primeiro deles é uma codificação de uma MT M e o segundo uma entrada x para esta MT (assim P interpreta as suas entradas). P retorna 1 se $M(x) \downarrow$ e 0 se $M(x) \uparrow$. Usando P, construimos a MT Q que toma dois inteiros m e x como parâmetros e:

(a) simula a execução de P com os parâmetros m e x;

(b) se P retorna 1, Q entra em um laço infinito, nunca terminando a sua execução;

(c) se P retorna 0, Q termina a sua execução.

O que acontece se chamamos Q passando $\langle Q \rangle$ e $\langle Q \rangle$ como parâmetros? Explique a contradição e o que podemos deduzir dela.

Resumo:

Uma expressão regular sobre um alfabeto Σ é descrita como: a) x é uma e.r. (expressão regular) se $x \in \Sigma$; b) ϵ 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 concatenação de duas linguagens L e K é $L \circ K = \{vw : v \in L \text{ e } w \in K\}$. E $L^\star = \{w_1 w_2 \ldots w_n | n \geqslant 0\}$.

O lema do bombeamento para linguagens regulares: se A é uma linguagem regular, então existe um inteiro p dependente de A tal que, se $s \in A$ e $|s| \geq p$, então s pode ser dividida em três partes, $s = xyz$, satisfazendo (a) $xy^iz \in A$ para todo $i \geq 0$; (b) $\left|y\right| > 0$ e (c) $\left|xy\right| \leq p$.

Uma máquina de Turing determinística (MT ou MTD) é uma quadrupla (Q, Σ, I, q) na qual Q e Σ 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, Σ 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.

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.

Dizemos que uma MT M enumera os elementos de uma linguagem $L \subset \Sigma^\star$ (ou $A \subset \mathbb{N}$) não vazia se:

(a) M despresa a sua entrada e imprime na fita um elemento de L, seguido de $\sqcup$, seguido de outro elemento de L, seguido de $\sqcup$ e assim por diante;

(b) dado $x \in L$, em algum momento da execução de M x será impresso na fita;

(c) M nunca pára a sua execução.

Note que os elementos impressos na fita por M podem ser repetidos e estar fora de ordem. Em particular, se L (ou A) for finito, haverá repetições.

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.

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

$$\mathrm{NP} = \bigcup_{k \in \mathbb{N}} \mathrm{NTIME}(n^k)$$