Prova Substitutiva de Teoria da Computação
Campus de Sorocaba da UFSCar
13 de dezembro de 2010
Prof. José de Oliveira Guimarães

Entregue apenas a folha de respostas. As questões não precisam estar em ordem e podem ser respondidas à lápis ou caneta. Na correção, símbolos ou palavras ilegíveis não serão considerados. Justifique todas as respostas a menos de menção em contrário.

Coloque o seu nome na folha de resposta, o mais acima possível na folha. Não é necessário colocar o RA. Se você não quiser que a sua nota seja divulgada publicamente, escreva apenas NÃO depois do seu nome.

No final da prova há um pequeno resumo da matéria.

Escolha uma e apenas uma questão entre 1 e 2.

1. (3.0) Converta a expressão regular $((11*) \cup (0*a))*b$ para um autômato finito. Faça a gramática que gera a mesma linguagem que esta expressão. A linguagem gerada pela expressão regular é regular. Que teorema garante isto? Enuncie-o.

2. (3,0) Dada a linguagem $L = \{0^n a^k 1^n : n, k \in \mathbb{N}\} \cup \{a0^k : k \geqslant 0\}$, faça ou responda:

   1. a gramática G tal que $L(G) = L$; isto é, a linguagem gerada pela gramática é igual a L;
   2. o autômato que a reconhece;
   3. no item anterior, poderia ser um autômato finito? Responda por que não sem fazer uma prova formal deste fato.

3. (3,0) O lema do bombeamento para linguagens regulares é o seguinte:

   Se A é uma linguagem regular, então há um número p no qual, se s é uma cadeia qualquer de A de tamanho $\geqslant p$, então s pode ser dividido em três pedaços, $s = xyz$ satisfazendo as seguintes condições:

   1. para cada $i \geqslant 0$, $xy^iz \in A$;
   2. $|y| > 0$, e
   3. $|xy| \leqslant p$.

   Explique, utilizando um autômato finito, porquê o lema do bombeamento é verdadeiro. Não é preciso provar o lema, apenas explique porquê ele funciona. Assuma que A é reconhecida por um autômato finito M com p estados. Não é necessário especificar completamete o autômato que é o exemplo.

Faça duas e apenas duas dentre as cinco questões seguintes

4. (2,5) Mostre como simular uma MT M com uma fita infinita nas duas direções por uma MT M' com uma fita infinita em apenas uma direção. Não apenas mostre a correspondência entre as fitas. Mostre como a simulação é feita (como é o algoritmo de M'). No início da execução, assuma que a cabeça de leitura e gravação de

(a) M está sobre o primeiro símbolo da entrada x (que está em binário);

(b) M' está sobre a segunda célula da fita (isto facilitará a simulação).

5. (3,0) Prove que a linguagem $K = \{<M>\sqcup x : M(x)\downarrow\}$ é recursivamente enumerável.

6. (2,5) Faça uma máquina de Turing que decida a linguagem $L = \{0^n 1^n : n > 1\}$. No início da execução, todas as células da fita, exceto a entrada, são brancos (caráter $\sqcup$). Assuma que exista algum outro símbolo no alfabeto da fita (pode precisar ...).

7. (3,0) Prove que a linguagem $H = \{<M>\sqcup x : M(x)\downarrow\}$ não é recursiva. Dica: assuma que H é decidida por um programa em C `int para(CM, x)` no qual `CM` é o código correspondente a M e `x` é a entrada.

8. (2,5) Prove que, se L e L^c são recursivamente enumeráveis, L é recursiva.

Resumo

Um autômato finito M é uma 5-tupla $(Q, \Sigma, \delta, q_0, F)$ no qual Q é um conjunto finito de estados, Σ é o alfabeto, $\delta : Q \times \Sigma \longrightarrow Q$ é a função de transição, $q_0 \in Q$ é o estado final e $F \subset Q$ e o conjunto de estados de aceitação.

Sendo R_1 e R_2 expressões regulares sobre Σ, uma expressão regular (e.r.) sobre um alfabeto Σ é definida indutivamente como: a) x é e.r. para $x \in \Sigma$ b) ϵ é e.r. c) $\emptyset$ é e.r. d) $(R_1 \cup R_2)$ é e.r. e) $(R_1 \circ R_2)$ é e.r. f) $(R^\star$ é e.r.

Sendo Σ um conjunto finito de símbolos, uma cadeia sobre Σ é a concatenação de um conjunto finito de símbolos de Σ. Definimos $\Sigma^n = \{a_1 a_2 \ldots a_n : a_i \in \Sigma, 1 \leqslant i \leqslant n\}$, o conjunto de todas as cadeias sobre Σ de tamanho n. Usamos ϵ para a cadeia com zero elementos. Logo $\Sigma 0 = \{\epsilon\}$. Definimos $\Sigma^\star$ como

$$\bigcup_{n \geqslant 0} \Sigma^n$$

Uma linguagem L sobre Σ é um subconjunto de $\Sigma^\star$.

Uma máquina de Turing M é uma 7-tupla $(Q, \Sigma, \Gamma, \delta, q_0, q_A, q_R)$ na qual Q, Σ, Γ são conjuntos finitos. Q é um conjunto de estados, Σ é o alfabeto de entrada ($\sqcup \in \Sigma$), Γ é o alfabeto da fita ($\sqcup \in \Gamma$ e $\Sigma \subset \Gamma$), $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R, S\}$ é a função de transição, $q_0 \in Q$ é o estado inicial, q_A é o estado de aceitação e q_R é o estado de rejeição. Sempre que o estado corrente for q_A ou q_R a máquina pára (e estes são os únicos estados finais). Para facilitar as provas, assuma que sempre que o estado corrente for q_A o valor de retorno da máquina será 1. Idem para q_R e 0.

Uma MT de decisão sempre termina o seu processamento e retorna 0 ou 1. A menos de menção em contrário, todas as MT aceitam um inteiro em binário como entrada. Uma MT M decide uma linguagem L sobre Σ se M é uma MT de decisão e $x \in L$ se e somente se $M(x) = 1$. Uma linguagem L sobre Σ é recursivamente enumerável se existe uma MT M tal que

$$x \in L \Longrightarrow M(x) = 1$$

$$x \notin L \Longrightarrow M(x) \uparrow$$

Uma linguagem $L \in$ NP se existe uma MT não determinística N que decide L em tempo polinomial; isto é, se $x \in L$, então existe uma sequência de escolhas não determinísticas na computação $N(x)$ de tal forma que o resultado seja $N(x) =^1$. Uma linguagem $L \in$ P se existe uma MT M que decide L em tempo polinomial. Isto é, $L \in \text{TIME}(n^k)$ para algum $k \in \mathbb{N}$. SAT é a linguagem $\{< \varphi >: \varphi$ está na FNC e é satisfazível $\}$. SAT é NP-completa. Isto é, $SAT \in NP$ e para toda $L \in$ NP, $L \leqslant_P SAT$ ($L \leqslant_P K$ se existe uma MT R que executa em tempo polinomial tal que $x \in L$ sse $R(x) \in K$). A relação $\leqslant_P$ é transitiva: se $L_1 \leqslant_P L_2$ e $L_2 \leqslant_P L_3$ então $L_1 \leqslant_P L_3$.

$A \sim B$ se existe uma função bijetora entre A e B.