Segunda Prova de Teoria da Computação — Campus de Sorocaba da UFSCar
Prof. José de Oliveira Guimarães, 31 de janeiro de 2013

As questões deverão ser respondidas nas folhas de respostas e apenas estas deverão ser entregues (não entregue a folha de questões). Utilize quantas folhas 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. (6,0) Faça os seguintes itens sobre máquinas de Turing (MT).

(a) (1,0) faça o diagrama de uma MT com uma única fita que escreve 1 na fita se a sua entrada é um número binário par e 0 caso seja ímpar. Assuma que a cabeça de leitura/gravação esteja posicionada na posição mais à esquerda da entrada. Não é preciso conferir se a entrada é realmente um número binário. À direita da entrada é colocado um símbolo $\square$. Ao final da execução, a célula corrente deve estar sobre a resposta;

(b) (1,0) qual o tempo de execução da MT do item anterior? Justifique precisamente a sua resposta, mostrando o que é o n da função de tempo de execução e o porquê da sua resposta;

(c) (1,0) faça a codificação da MT que possui as seguintes instruções: $(q_0, 0, q_0, 0, P), (q_0, 1, q_0, 0, D)$. Codificação é transformar a MT em um número inteiro de tal forma que, dado um $n \in \mathbb{N}$, pode-se saber de n codifica alguma MT ou não. Se sim, pode-se recuperar toda a descrição da MT por meio de n. O alfabeto utilizado é $\Sigma = \{0, 1, \triangleright, \sqcup, \square\}$ e há apenas q_0 de estado;

(d) (1,0) cite a tese de Church-Turing;

(e) (0,5) cite um outro mecanismo de se definir "mecanicamente computável"? Apenas cite. Não vale variações da máquina de Turing;

(f) (1,0) explique o que é uma máquina de Turing Universal. Explique claramente que parâmetros ela toma. Ela sempre para?

2. (2,0) Suponha que exista uma MT P que toma a codificação $\langle M \rangle$ (um inteiro) de uma MT M e um inteiro x como parâmetros e retorna 0 se $M(x)$ retorna 0 e 1 se $M(x)$ retorna algum outro valor ou não termina a computação. Utilizando, obrigatoriamente, o teorema da recursão, faça uma MT $S(x)$ tal que S dá origem a uma contradição. Deixe bem claro qual é a contradição.

A descrição de S deve ser feita com palavras. Mas cuidado: você pode dizer que uma MT retorna um valor (como na apostila) mas uma MT não pode "chamar" uma outra. Mas pode-se usar uma MT Universal ou simular uma outra máquina da qual conhecemos as instruções.

3. (2,5) Sobre máquinas de Turing não determísticas (MTND) e máquinas de Turing determinísticas (MTD), responda:

(a) (1,0) sendo Q um conjunto de estados, Σ um alfabeto, $D = \{E, P, D\}$ a direção do movimento da cabeça de leitura/gravação, especifique as funções de transição δ da MTND e da MTD. Algo assim: $\delta : \Sigma \longrightarrow Q \times \Sigma$ ou como relação: $\delta \subset \Sigma \times Q \times \Sigma$;

(b) (1,5) explique a diferença entre a execução de um único passo de uma MTND e uma MTD, sem entrar em detalhes do comportamento geral ou da diferença das duas em relação à aceitação de uma cadeia ou não. Obrigatoriamente cite dois exemplos na sua resposta, uma de uma MTND e outra de uma MTD, comparando-os. A resposta deve ser suscinta.

Faça uma e apenas uma das três questões seguintes.

4. (2,5) Se L e L^c são linguagens recursivamente enumeráveis, então L e L^c são recursivas.

5. (2,5) Prove que a cardinalidade do conjunto T de todas as máquinas de Turing, $|T|$, é menor ou igual à cardinalidade de $\mathbb{N}$, $|\mathbb{N}|$. Isto é, prove que há uma função injetora entre T e $\mathbb{N}$.

6. (2,5) Prove que a linguagem $H = \{\langle M \rangle \sqcup x : M(x) \downarrow\}$ é recursivamente enumerável. Isto é, existe uma MT N tal que $N(\langle M \rangle \sqcup x)$ retorna 1 se $M(x) \downarrow$ (isto é, $\langle M \rangle \sqcup x \in H$). E, se $M(x) \uparrow$ (isto é, $\langle M \rangle \sqcup x \notin H$), então $N(\langle M \rangle \sqcup x)$ entra em laço infinito, $N(\langle M \rangle \sqcup x) \uparrow$.

7. (10^{100}) P = NP?

Resumo A menos de menção em contrário, todas as entradas para uma MT é um número inteiro em binário.

Usamos $\langle M \rangle$ para a codificação da MT M e $M(x)$ para o resultado que a MT M produz com entrada x. Se M não parar a sua execução, escrevemos $M(x) \uparrow$. Se parar, podemos escrever $M(x) \downarrow$. Um enumerador é uma MT com duas fitas que ignora a sua entrada e escreve números na segunda fita, separados por brancos. Os números da segunda fita fazem parte de um conjunto recursivamente enumerável.

Seja S o conjunto de todas as linguagens sobre $\{0, 1\}$, $2^{\mathbb{N}}$ o conjunto de todas as funções de $\mathbb{N}$ em $\{0, 1\}$, $\mathbb{N}^{\mathbb{N}}$ o conjunto de todas as funções de $\mathbb{N}$ em $\mathbb{N}$, T o conjunto de todas as máquinas de Turing e T_D o conjunto de todas as MT de decisão (a cada uma delas está associada uma linguagem).

Uma linguagem L é recursiva ou computável 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 recursivamente enumerável (r.e.) ou computacionalmente enumerável 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.

O teorema da recursão diz que uma MT M pode ser transformada em uma MT R de tal forma que R faz exatamente o que M faz mas R também tem acesso ao seu próprio código. Então R sabe quantos estados R tem, R pode simular R e assim por diante.

Dizemos que uma MT M executa em tempo $f(n)$ se, para entradas de tamanho n, o número de passos que ela toma é menor ou igual a $f(n)$.