Universidade Federal do Rio Grande do Sul	Instituto de Informática
INF05501 – Teoria da Computação	2019/1
Frederico Messa Schwartzhaupt	00304244
Henry Bernardo Kochenborger de Avila	00301161
Marcos Samuel Winkel Landi	00304688

Lista de Exercícios 2

Questão 1: (2 pontos)

Exatamente um dos problemas abaixo é semi-decidível. Diga qual problema é/não é semi-decidível justificando sua resposta.

> **Instância:** Uma máquina de Turing M.
> **Pergunta:** $ACEITA(M)$ contém pelo menos 20 palavras?

> **Instância:** Uma máquina de Turing M.
> **Pergunta:** $ACEITA(M)$ contém menos do que 20 palavras?

Usando o teorema de Rice, temos que, toda propriedade não-trivial sobre uma linguagem enumerável recursivamente é indecidível. Logo, é possível verificar que a propriedade sobre a linguagem do primeiro problema é ela conter pelo menos 20 palavras e ela é não-trivial - assim como a propriedade do segundo problema que, por sinal, é o complemento do primeiro.

De fato, é possível verificar que existem linguagens que não contém essa propriedade - a linguagem vazia, por exemplo - e existem linguagens que contém - a linguagem que contém todos os símbolos do sistema de numeração vigesimal.

Portanto, como a propriedade é não-trivial, então o problema é indecidível.

Visto isso, é possível que exista uma máquina de Turing T que recebe uma máquina de Turing M e realiza o seguinte processo:

1. Atribue a c o valor 0, a t o valor 0 e a l a lista vazia.
2. Acrescente a l simulações de M com todas as palavras de tamanho t.
3. Avance um passo em todas simulações existentes.
4. Veja se alguma simulação parou. Se sim, exclua ela da lista de simulações existentes (l) e, se parou aceitando, incremente c.
5. Verifique se $c = 20$. Se sim, pare aceitando M. Senão, incremente t e volte ao passo 2.

Dessa maneira, a máquina T sempre para para qualquer entrada M tal que $|ACEITA(M)| \geq 20$, pois como essas palavras têm tamanho finito e a máquina T verifica todas as palavras de tamanho finito como entrada para essas máquinas, então T irá parar aceitando. Ademais, T entra em loop sempre que receber uma máquina M' tal que $|ACEITA(M')| < 20$. Portanto, como T é uma máquina que semi-decide o primeiro problema, então o primeiro problema é semi-decidível.

Considerando que os problemas são exatamente o complemento um do outro, visto que, se, dada uma máquina de Turing M tal que $|ACEITA(M)| < 20$, então $P_1(M) = false$ e $P_2(M) = true$. Já, se $|ACEITA(M)| \geq 20$, então $P_1(M) = true$ e $P_2(M) = false$.

Assim sendo, como P_1 é indecidível e é semi-decidível, então P_2 deve ser completamente insolucionável, pois, se fosse semi-decidível, P_1 seria decidível, o que é falso pelo teorema de Rice.

Questão 2: (2 pontos)

Para cada um dos problemas abaixo, diga se o problema é decidível ou não justificando sua resposta.

- **(1 ponto)** PROBLEMA DA PARADA COM ESPAÇO QUADRÁTICO
  Instância: Uma máquina de Turing M e uma palavra w.
  Pergunta: M para com entrada w gastando não mais do que $|w|^2$ de espaço?

- **(1 ponto)** PROBLEMA DA EXPANSÃO DECIMAL DE e
  Instância: Um número natural $n \geq 1$
  Pergunta: Existe uma sequência contígua com pelo menos n 9's na expansão decimal do número de Euler e?

- **Problema da parada com espaço quadrático:**

  Se esse problema é decidível, então existe uma máquina de Turing MT que recebe um par (M, w), tal que M é uma máquina de Turing e w é uma palavra, e que nunca entra em loop - ou seja, devolve sim ou não para qualquer entrada.

  Sendo assim, MT pode ser dada por uma máquina de Turing com duas fitas que faz os seguintes processos:

  1. Simula M para a palavra w na primeira fita, sendo que, a cada passo, verifica se o uso da fita ultrapassou $|w|^2$. Se sim, MT rejeita a entrada. Senão, vai para o passo 2.

2. Verifica se M parou (está em um estado final em M ou rejeita por indefinição). Se sim, aceita a entrada. Senão, escreve a configuração atual da máquina (o estado em que se encontra, a fita 1 até $|w|^2$ e a posição atual da cabeça na fita 1) em alguma codificação na fita 2 e compara com as outras presentes na fita 2. Se existir uma outra configuração idêntica na fita 2, então significa que a máquina entrou em loop e a entrada é rejeitada. Senão, volta para o passo 1.

Isso se dá porque a máquina é determinística, ou seja, dado um estado, uma fita de uma maneira específica e a cabeça numa certa posição, o próximo passo será sempre o mesmo. Dessa forma, como o número de estados é finito e o número de possíveis situações da fita também - visto que, se ultrapassar $|w|^2$ a máquina já para no passo 1 - então o número de combinações é finito e em algum momento a máquina irá parar ou terá chegado no número máximo de combinações de estado atual e fita, e irá gerar no próximo passo uma combinação que já está presente na fita 2, rejeitando a entrada.

Portanto, MT decide o problema e o problema é decidível.

- **Problema da expansão decimal de e:**

  Uma das possibilidades para o conjunto de instâncias codificadas do problema é a linguagem $L = \{1^n | n \in \mathbb{N}^*\}$. Dessa forma, suponha que o problema seja decidível. Assim sendo, existe uma máquina de Turing M que recebe uma instância do problema da expansão decimal de e e para para qualquer entrada. Considerando isso, existem duas possibilidades para $ACEITA(M)$:

  – $ACEITA(M)$ **é finita:** dessa forma, é importante notar que, para qualquer linguagem finita, é possível criar uma máquina de Turing que verifica se a palavra de entrada é alguma palavra pertencente a essa linguagem. Assim sendo, o conjunto A, dado por todas as máquinas de Turing que contém alguma combinação finita c dos números naturais como palavras pertencentes às linguagens de aceitação e $\forall w \in L, MT_c \in A | w \in ACEITA(MT_c) \iff w \in c$, com certeza contém a máquina M, pois a máquina M nada mais é que uma máquina de Turing que verifica se a palavra de entrada é igual a alguma palavra de uma certa combinação de palavras (essas que são pertencentes a linguagem L). Portanto, M existe e decide o problema da expansão decimal de e.

  – $ACEITA(M)$ **é infinita:** consequentemente, $ACEITA(M) = L$. Para isso, usarei uma prova por contradição.

  Suponha que $ACEITA(M) \neq L$. Assim sendo, como $L = \{1^n | n \in \mathbb{N}^*\}$, então deve existir uma palavra $w_n \in L$ que se refere a um número n tal que $w_n \notin ACEITA(M)$. Dessa forma, como a sequência deve ser contígua, então $(\forall w_m \in L) \wedge (m \geq n) \implies w_m \notin ACEITA(M)$, pois, se $w_m \in ACEITA(M)$, então a "subsequência" de tamanho

n estaria contida na sequência de tamanho m, mas como $w_n \notin ACEITA(M)$, então essa afirmação é verdadeira. Contudo, se essa afirmação for verdadeira, então $ACEITA(M)$ deve ser finita, pois, para quaisquer dois números naturais a e b tal que $a \leq b$, o conjunto formado por todos os números naturais a e b é necessariamente um conjunto finito. Contradição, pois a suposição inicial é que $ACEITA(M)$ é infinita. Logo, $ACEITA(M) = L$. Como L é a linguagem que codifica os números naturais positivos, então é possível construir uma máquina de Turing T que decide L simplesmente verificando se a palavra de entrada é uma sequência de 1's. Por conseguinte, $M = T$ e o problema da expansão decimal de e é decidível.

Como em todos os casos o problema é decidível, então o problema é decidível.

Questão 3: (2 pontos)

Prove que o problema descrito abaixo não é semi-decidível. Prove também que o *complemento* do problema descrito abaixo não é semi-decidível.

> – PROBLEMA DAS LINGUAGENS DE ACEITAÇÃO IDÊNTICAS
> **Instância:** Máquinas de Turing M_1 e M_2.
> **Pergunta:** É verdade que $ACEITA(M_1) = ACEITA(M_2)$?

Para provar que esse problema não é semi-decidível, irei utilizar um corolário.

- **Corolário:** sejam A e B problemas de decisão. Se A não é semi-decidível e existe uma redução $r : A \to B$, então B não é semi-decidível.

  Prova por contradição:
  Se A não é semi-decidível, existe uma redução $r : A \to B$ e B é semi-decidível, então é possível utilizar, para qualquer instância de a de A, $M_B(r(a))$ - tal que M_B é a máquina de Turing que semi-decide B - para semi-decidir A. Dessa forma, A é semi-decidível. Contradição!

Assim sendo, levando em consideração que esse problema ($PA - IGUAL$) é indecidível (de acordo com o slide 384). Então só existem três possibilidades:

- $PA - IGUAL$ é semi-decidível e $\overline{PA - IGUAL}$ não é semi-decidível.
- $\overline{PA - IGUAL}$ é semi-decidível e $PA - IGUAL$ não é semi-decidível.
- $PA - IGUAL$ não é semi-decidível e $\overline{PA - IGUAL}$ não é semi-decidível.

Portanto, se existe uma redução $r : PA - IGUAL \to \overline{PA - IGUAL}$ e existe uma redução $t : \overline{PA - IGUAL} \to PA - IGUAL$, então ambos problemas não são semi-decidíveis, pois os dois primeiros casos implicam no terceiro caso.

Dessa forma, seja (M, M') uma instância de $PA{-}IGUAL$, então $r((M, M')) =$ ()

Questão 4: (2 pontos)

Seja $M = (\Sigma, Q, \Pi, q_0, F, V, \beta, ©)$ uma máquina de Turing. Dizemos que uma transição de M é *corretiva* se tal transição faz com que M escreva na fita seu símbolo de espaço em branco β por cima de qualquer símbolo que seja diferente de β. Por exemplo, sejam $q, q' \in Q$ e $a \in \Sigma$. Suponha que $\Pi(q, a) = (q', \beta, D)$. Como $a \neq \beta$ (pois, por definição, $\beta \notin \Sigma$), então a transição de M descrita por $\Pi(q, a)$ é uma transição corretiva, visto que tal transição ordena que M escreva o símbolo de espaço em branco β em cima de um símbolo diferente de β (nesse caso a).

Mostre que, para *toda* máquina de Turing M, existe uma máquina de Turing M' *equivalente* a M (veja no final do enunciado desta questão o conceito de máquinas de Turing equivalentes) *sem transições corretivas*. Em outras palavras, esta questão, essencialmente, pede que você prove que "proibir" transições corretivas em máquinas de Turing não implica na diminuição do seu poder computacional.

É simples fazer a contrução de M', pois basta acrescentar uma letra a que não pertence a $\Sigma \cup V$ ao alfabeto auxiliar e, nas transições corretivas, trocar o símbolo de espaço em branco por essa letra. Ademais, para toda transição t que existir em M e ler da fita o símbolo do espaço em branco, acrescente uma transição t' que faça exatamente a mesma coisa (saia do mesmo estado que t sai, vai para o mesmo estado t vai, escreve na fita a mesma letra que t escreve e move a cabeça no mesmo sentido que t move) só que lê da fita a e, se t escreve o espaço em branco em cima do espaço em branco lido, então t' escreve a em cima do a lido. Assim sendo, é acrescentado outro símbolo que é equivalente ao símbolo de espaço em branco, mas não é ele, e todas as transições corretivas são retiradas.

Dessa forma, é possível verificar que para qualquer M, existe uma M' equivalente, pois as transições corretivas foram trocadas por outras transições equivalentes quanto ao reconhecimento de linguagens. E como a máquina de Turing é, essencialmente, um reconhecedor de linguagens, então essa equivalência computacional é válida.

Já, para computação de funções, seria necessário modificar como as funções são computadas, sendo que a função que a apaga da fita os espaços vazios também iria apagar da fita esse símbolo extra acrescentado.

Questão 5: (2 pontos)

Prove que se uma função total $f : \mathbb{N} \to \mathbb{N}$ é bijetora e Turing computável (isto é, pode ser computada por uma máquina de Turing), então f^{-1} (isto é, a inversa de f) é total e Turing-computável.

Suponha que $f : \mathbb{N} \to \mathbb{N}$ é uma função total, bijetora e Turing computável. Logo, como é total, então ela também é Turing decidível, pois, para toda entrada, existirá uma saída. Ou seja, $LOOP(M)$, tal que M é a máquina de Turing que computa essa função, é a linguagem vazia.

Assim sendo, é possível definir $f^{-1} : \mathbb{N} \to \mathbb{N}$ por uma máquina de Turing que recebe um número natural e testa todas as entradas possíveis para f, iniciando em 0, verificando se a saída de f é igual ao número natural dado como entrada. Se for, devolve a entrada dada pra f nessa iteração. Senão, incrementa a entrada de f e realiza o teste novamente. Dessa forma, f^{-1} é Turing computável.

Já para verificar que f^{-1} é total, irei provar que para qualquer função total bijetora $\delta : A \rightarrow B$, existe uma função inversa $\delta^{-1} : B \rightarrow A$ tal que δ^{-1} é total. Para isso, segue uma prova por contradição.

- Suponha que δ^{-1} seja a inversa de δ e não é total. Logo, $\exists b \in B, \forall a \in A | \delta^{-1}(b) \neq a$. Agora, como δ é bijetora, então $\exists c \in A | \delta(c) = b$. Assim sendo, como $\forall a \in A, \delta^{-1}(\delta(a)) = a$ - visto que δ^{-1} é a inversa de δ -, então $\delta^{-1}(\delta(c)) = c = \delta^{-1}(b)$. Contradição, pois $\forall a \in A | \delta^{-1}(b) \neq a$. Logo, δ^{-1} é total.

Dessa forma, f^{-1} também é total.