

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2213 — Lógica para Ciencia de la Computación — 2024-01

TAREA 3

Publicación: Lunes 22 de abril.
Entrega: **Martes 14 de mayo hasta las 23:59 horas**.

Indicaciones

- La tarea es **individual**. Puede discutir con sus compañera/os o utilizar información externa (libros, foros, páginas web, etc...), siempre y cuando usted mismo escriba su tarea, con sus propias palabras. Debe **citar** cualquier referencia que utilice. Se considerará copia: usar soluciones de compañera/os (con o sin consentimiento del plagiado), entregar su solución para que la vea otra/o compañera/o, usar fuentes externas sin citar.
- Cada pregunta tiene una nota entre 1 y 7 (hay un punto base). Los items dentro de una pregunta valen lo mismo. La nota de la tarea es el promedio de las notas de las preguntas.
- La tarea **debe** ser escrita en LaTeX. No se aceptarán tareas escritas de otra forma. No olvide indicar su nombre, RUT y número de alumna/o.
- Se aceptan atrasos. Se descontará 1 punto por día de atraso, sobre la nota final. Se permite como máximo 3 días de atraso.

Pregunta 1

Considere el modelo de máquinas de Turing con múltiples cabezas lectoras descrito al final de la Clase 05. Demuestre que para toda MT M con k cabezas lectoras, existe una MT normal M' tal que $L(M') = L(M)$.

Observación: Basta con que describa el funcionamiento de alto nivel de la MT M'. (por ejemplo, como lo hicimos en clases para MTs con múltiples cintas).

Solución

Dada una máquina M con múltiples cabezas lectoras $M = (Q, A, \Lambda, q_0, F, \delta)$, definiremos una MT normal M' que la simula. Para simular un paso de M desde un estado q, la máquina M' hace:

1. Tenemos almacenado q y la cabeza lectora está al inicio de la palabra.
2. Escaneamos la cinta de izquierda a derecha recordando el símbolo que lee cada cabeza (esta información se almacena en los estados de M').
3. Una vez vistos estos k símbolos $a_1, \ldots, a_k$, escaneamos la cinta de derecha a izquierda actualizando la cinta según la transición $\delta(q, a_1, \ldots, a_k)$ de M.
4. Al terminar, volvemos al inicio de la cinta, almacenamos el nuevo estado según $\delta(q, a_1, \ldots, a_k)$, y simulamos el siguiente paso.

Intuitivamente, la idea es que vamos marcando dónde está cada cabeza lectora y luego usamos esa información para ir ejecutando la transición en cada una.

Dar el puntaje completo en caso que se presente una máquina M' que simule M correctamente, aunque no sea la misma presentada aquí. En caso que haya algún error pequeño en la ejecución que sea fácilmente solucionable, bajar 1 pt. Si se presenta una máquina que simula M' de manera completamente incorrecta, se da solo 1 pt.

Pregunta 2

1. Demuestre que L es decidible si y sólo si existe una MT enumeradora E que enumera L en orden lexicográfico (i.e., si E imprime w' *después* de w, se tiene que w' es *lexicográficamente mayor* que w).

   Dado un alfabeto finito $A = \{a_1, \ldots, a_k\}$, donde asumimos un orden $a_1 < a_2 < \ldots < a_k$, $w' = w'_1 \ldots w'_p$ es *lexicográficamente mayor* que $w = w_1 \cdots w_q$ si $p > q$, o $p = q$ y $w'_i > w_i$, donde $i \in \{1, \ldots, p\}$ es la primera posición tal que $w'_i \neq w_i$.

   Por ejemplo, si $A = \{a, b, c, d\}$, donde $a < b < c < d$, las siguientes palabras están en orden lexicográfico, de menor a mayor: ε, a, b, d, aa, ab, bc, aba, bca, bda, bdd, cca, daa, $aaaa$.

2. Demuestre que todo lenguaje infinito recursivamente enumerable L tiene un subconjunto $L' \subseteq L$ tal que L' es infinito y decidible. (Hint: use el punto anterior.)

Solución

Parte 1

($\Rightarrow$) Sea L un lenguaje decidible, entonces definimos la máquina E de la siguiente manera:

1. En la cinta de trabajo se comienza escribiendo el símbolo menor.
2. Se ejecuta M en la cinta de trabajo.
3. Si M acepta, se copia el input en la cinta enumeradora.
4. Se escribe el siguiente input en orden lexicográfico en la cinta de trabajo. Para esto se escribe el siguiente símbolo en la última posición de la palabra y en caso que este haya sido el símbolo mayor, se escribe el símbolo menor, se lee el penúltimo símbolo y se repite lo anterior.
5. Se vuelve al paso 2.

Como L es decidible, sabemos que toda ejecución de M será finita, por lo tanto, E efectivamente podrá enumerar todo el lenguaje sin quedarse pegada.
($\Leftarrow$) Para esta implicancia hay que considerar dos casos:

1. L es finito: Todo lenguaje finito es decidible. En particular, L es decidible.
2. L es infinito: Si tenemos una máquina enumeradora E para L, entonces podemos construir una máquina de tres cintas M de la siguiente manera:

   a) M recibe un input w en la primera cinta
   b) Se ejecuta E en la segunda y tercera cinta.
   c) Cada vez que se copia una nueva palabra a la cinta enumeradora se revisa si es igual a w, si es así acepta. Si la nueva palabra es lexicográficamente mayor a w, entonces se rechaza.

   Esta máquina acepta a L, ya que si $w \in L$ eventualmente será escrita en la cinta enumeradora y si $w \notin L$ también se detiene, ya que existen palabras lexicográficamente mayores a w en L por la infinitud del lenguaje.

Parte 2

Sea L un lenguaje infinito recursivamente enumerable, entonces por un teorema visto en clases (slide 20 de la clase 6), sabemos que existe una máquina E que enumera a L, pero no necesariamente en orden lexicográfico. Luego, construimos una nueva máquina E' que simula E y cada vez que se agrega una nueva palabra w a la cinta enumeradora, se revisa si efectivamente es lexicográficamente mayor que la palabra anterior y si no es así, se borra de la cinta.
Claramente el lenguaje que se va escribiendo será un subconjunto de L y estará en orden lexicográfico, lo que falta comprobar es que efectivamente sea infinito. Para esto, notemos que al ser L infinito, contiene palabras arbitrariamente grandes en orden lexicográfico, por lo tanto, si existiera un w tal que no tiene ninguna palabra lexicográficamente mayor que sea escrita por E después de w, entonces tendríamos que haber enumerado infinitas palabras antes de w. $\multimap$
En conclusión, existe una máquina E' que enumera un subconjunto infinito L' de L en orden lexicográfico. Por parte 1 de esta pregunta, esto quiere decir que ese subconjunto es un lenguaje decidible.

Cada parte tiene 3 pts. En la primera se da 1.5 pts. por cada lado de la implicancia y en cada implicancia se da 1 pt. por la construcción de la MT correspondiente y 0.5 pts. por la justificación. En caso de no considerar L finito en la segunda implicancia, no descontar puntaje, pero agregar un comentario a la corrección.
En la parte 2, se otorgan 2 puntos por construir correctamente la MT y 1 pt. por justificar adecuadamente por qué L' es un subconjunto infinito decidible de L.

Pregunta 3

1. Considere el siguiente lenguaje:

$$L = \{\text{cod}(M) \cdot w \mid M \text{ es una MT}, w \in \{0,1\}^*, M \text{ se detiene sobre } w \text{ en 100 o más pasos}\}.$$

   Mediante una reducción desde el problema Halting, visto en clases, demuestre que L es indecidible.

2. En clases vimos que el siguiente lenguaje es indecidible:

$$L = \{\text{cod}(M) \mid M \text{ es una MT y } L(M) \neq \emptyset\}.$$

   Mediante una reducción desde L, demuestre que el siguiente lenguaje es indecidible:

$$L' = \{\text{cod}(M_1) \cdot \text{cod}(M_2) \mid M_1 \text{ y } M_2 \text{ son MTs y } L(M_1) = L(M_2)\}.$$

Solución

Parte 1

Definimos la reducción $f : \{0,1\}^* \to \{0,1\}^*$ de Halting a L de la siguiente manera:

1. Si z no tiene la forma $z = \text{cod}(M) \cdot w$ entonces $f(z) = 0 \notin L$.
2. Si $z = \text{cod}(M) \cdot w$ para alguna MT M y $w \in \{0,1\}^*$, entonces

$$f(\text{cod}(M) \cdot w) = \text{cod}(N_M) \cdot w$$

   donde N_M es una máquina tal que para toda entrada $u \in \{0,1\}^*$:

   a) Se avanza 50 posiciones a la derecha y luego 50 posiciones a la izquierda.

   b) Se simula M sobre u y se acepta cuando M acepta o rechaza u.

Probemos que esta función efectivamente es una reducción, es decir, $z \in$ Halting si y solo si $f(z) \in L$.
($\Rightarrow$) Si $z \in$ Halting, entonces $z = \text{cod}(M) \cdot w$ y M es una máquina que se detiene cuando tiene a w como input, es decir, lo acepta o lo rechaza en $n \in \mathbb{N}$ pasos. Luego, N_M se detiene en $100 + n \geq 100$ pasos y $\text{cod}(N_M) \cdot w \in L$.
($\Leftarrow$) Si $z \notin$ Halting, entonces podría ser que z no está en el formato adecuado, en cuyo caso $f(z) = 0 \notin L$, o $z = \text{cod}(M) \cdot w$, pero la ejecución de M sobre w nunca se detiene. En el segundo caso, tenemos que N_M actuará igual que M sobre w, solo se agregan 100 pasos al comienzo, entonces tampoco detendrá su ejecución, en particular, no se detiene en 100 o más pasos, es decir $f(z) \notin L$.
Concluimos así que la reducción presentada cumple lo pedido y como Halting es indecidible, sabemos que L también será indecidible.

Parte 2

La reducción para esta pregunta se puede hacer desde el lenguaje sugerido en el enunciado o desde Halting. Aquí se presentarán ambas opciones:

Reducción desde L

Definimos la reducción $f : \{0,1\}^* \to \{0,1\}^*$ de L a L' de la siguiente manera:

1. Si z no tiene la forma $z = \text{cod}(M)$ entonces $f(z) = 0 \notin L'$.

2. Si $z = \text{cod}(M)$ para alguna MT M, entonces

$$f(\text{cod}(M)) = \text{cod}(N_M) \cdot \text{cod}(M_{\text{all}})$$

donde N_M es una máquina que acepta todo input si existe alguna palabra aceptada por M y M_{all} es una máquina que acepta todo input.

Para construir la máquina N_M se puede utilizar la misma idea de la última diapositiva de la clase 6. Es decir, construimos una máquina tal que

1. N_M borra el input.
2. N_M itera sobre todos los pares $(w, 0^i) \in A^* \times \{0\}^*$.
3. Para cada par $(w, 0^i)$ se simulan i pasos de M sobre w
4. Si M acepta algún w, entonces N_M acepta.

Esta máquina aceptará al encontrar algún input aceptado y eventualmente itera sobre todo $A^* \times \{0\}^*$, por lo tanto si no acepta significa que M no acepta ninguna palabra.
Luego, solo queda probar que $z \in L$ si y solo si $f(z) \in L'$.
($\Rightarrow$) Si $z \in L$, entonces $z = \text{cod}(M)$ tal que $L(M) \neq \emptyset$, por lo tanto N_M aceptará todo input, es decir $L(N_M) = L(M_{\text{all}})$ y $f(\text{cod}(M)) = \text{cod}(N_M) \cdot \text{cod}(M_{\text{all}}) \in L'$
($\Leftarrow$) Si $z \notin L$, entonces z no está en el formato adecuado y $f(z) \notin L'$ o $z = \text{cod}(M)$ para alguna máquina M que tiene lenguaje vacío, en cuyo caso N_M no se detendrá nunca para ningún input, por lo que $L(N_M) = \emptyset$ y tendremos que $L(N_M) \neq L(M_{\text{all}})$. Es decir, $f(z) \notin L'$.
Concluimos así que la reducción presentada cumple lo pedido y como L es indecidible, sabemos que L' también será indecidible.

Reducción desde Halting

Definimos la reducción $f : \{0,1\}^* \to \{0,1\}^*$ de Halting a L' de la siguiente manera:

1. Si z no tiene la forma $z = \text{cod}(M)$ entonces $f(z) = 0 \notin L'$.
2. Si $z = \text{cod}(M) \cdot w$ para alguna MT M y $w \in \{0,1\}^*$, entonces

$$f(\text{cod}(M) \cdot w) = \text{cod}(N_{(M,w)}) \cdot \text{cod}(M_{\text{all}})$$

donde M_{all} es la misma de la reducción anterior y $N_{(M,w)}$ es una máquina tal que

a) Dado cualquier input, $N_{(M,w)}$ lo borra y escribe w.
b) Se simula la ejecución sobre de M sobre w. Si M acepta o rechaza, entonces $N_{(M,w)}$ acepta.

Luego, solo queda probar que $z \in$ Halting si y solo si $f(z) \in L'$.
($\Rightarrow$) Si $z \in$ Halting, entonces $z = \text{cod}(M) \cdot w$ tal que al ejecutar M sobre w, esta se detiene, es decir acepta o rechaza. Por lo tanto, $N_{(M,w)}$ aceptará siempre y $L(N_{(M,w)}) = L(M_{\text{all}})$, i.e. $f(z) \in L'$.
($\Leftarrow$) Si $z \notin L$, entonces z no está en el formato adecuado y $f(z) \notin L'$ o $z = \text{cod}(M) \cdot w$ pero la ejecución de M sobre w nunca se detiene. En este caso, se tendría que la ejecución de $N_{(M,w)}$ tampoco se detiene nunca, entonces no acepta nungún input, por lo que $L(N_{(M,w)}) = \emptyset \neq L(M_{\text{all}})$ y $f(z) \notin L'$.
Concluimos así que la reducción presentada cumple lo pedido y como Halting es indecidible, sabemos que L' también será indecidible.

Cada parte tiene 3 puntos, 1.5 pts. se dan por hacer la reducción correctamente y los otros 1.5 pts. se dan por justificar adecuadamente (deben presentar ambas direcciones de la doble implicancia). No es necesario que mencionen ni justifiquen computabilidad.