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

IIC2213 — Lógica para ciencia de la computación — 1' 2023

TAREA 3

Publicación: Miércoles 12 de abril.
Entrega: **Viernes 28 de mayo hasta las 23:59 horas**.

Indicaciones

- Cada pregunta tiene 6 puntos (+1 base) y la nota de la tarea es el promedio de las preguntas.
- La solución debe estar escrita en LaTeX. No se aceptarán tareas escritas de otra forma.
- La tarea es individual, pudiendo discutirla con sus pares. Toda referencia externa debe citarse.

Objetivos

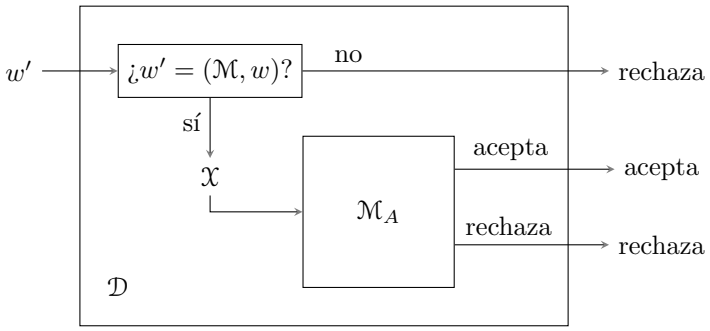
- Demostrar que lenguajes son indecidibles usando máquinas y reducciones.
- Identificar lenguajes R y RE y demostrar propiedades.
- Construir reducciones entre diferentes lenguajes.

Pregunta 1: Sintaxis e inducción en $\mathcal{L}(P)$

(a) Considere el siguiente lenguaje

$$A = \{(\mathcal{M}, w) \mid \mathcal{M} \text{ es una TM determinista tal que } \mathcal{M} \text{ acepta } w\}$$

Note que este lenguaje es similar, pero no idéntico a H visto en clases, pues A habla de aceptar w y no solo de detenerse como lo hace H. Para demostrar que A es indecidible, supongamos que es decidible y por ende existe una máquina $\mathcal{M}_A$ que lo decide. Usando $\mathcal{M}_A$, se puede construir una máquina $\mathcal{D}$ que decide H, obteniendo una contradicción. A continuación se presenta la estructura de $\mathcal{D}$ que decide H



(i) Dé una descripción de la máquina que queda codificada en $\mathcal{X}$ y que se alimenta a $\mathcal{M}_A$. Puede ser un diagrama o una descripción con palabras. Si hace un diagrama, puede hacerlo a mano e incluirlo como foto, no es necesario usar la librería `tikz`.

(ii) Demuestre que la máquina $\mathcal{D}$ decide H.

(b) Un lenguaje indecidible conocido es

$$L_1 = \{(\mathcal{M}) \mid \mathcal{M} \text{ es una TM determinista tal que } L(\mathcal{M}) = \varnothing\}$$

Mediante reducción desde L_1, demuestre que el siguiente lenguaje también es indecidible

$$L_2 = \{(\mathcal{M}_1, \mathcal{M}_2) \mid \mathcal{M}_1, \mathcal{M}_2 \text{ son TM deterministas tales que } L(\mathcal{M}_1) = L(\mathcal{M}_2)\}$$

Solución

Parte (a)

(i) Definimos $\mathcal{X}$ como la máquina tal que si $\mathcal{M}$ acepta w $\mathcal{X}$ la acepta y si $\mathcal{M}$ rechaza w, entonces $\mathcal{X}$ lo acepta. Esto se puede, ya que la máquina se detiene cuando acepta o rechaza.

(ii) Sea w' un input para $\mathcal{D}$. Tenemos tres casos posibles, w' no tiene el formato $(\mathcal{M}, w)$, w' tiene el formato adecuado pero no se detiene o w' tiene el formato adecuado y se detiene. Demostremos que $\mathcal{D}$ solo acepta los inputs del último caso:

1. Si w' no tiene el formato adecuado, es inmediatamente rechazado.
2. Si $w' = (\mathcal{M}, w)$ y no se detiene, entonces $\mathcal{X}$ no se detiene con el input w', en particular no acepta el input w', por lo tanto, $\mathcal{M}_A$ rechazará, es decir $\mathcal{D}$ rechaza w'.
3. Si $w' = (\mathcal{M}, w)$ y no se detiene, entonces $\mathcal{M}$ acepta o rechaza w, por lo tanto $\mathcal{X}$ aceptará w y $\mathcal{M}_A$ aceptará el input, por lo que $\mathcal{D}$ acepta.

Por lo tanto $L(\mathcal{D}) = H$ y esta máquina se detiene siempre, por lo que llegamos a la contradicción pedida.

Parte (b)

Definiremos la función $f : \{0,1\}^* \to \{0,1\}^*$ como la función tal que si $w \in \{0,1\}^*$ no es la codificación de una máquina de Turing, entonces $f(w) = 0$ y en caso que $w = (\mathcal{M})$, entonces $f(w) = (\mathcal{M}, \mathcal{M}_\varnothing)$ donde $\mathcal{M}_\varnothing$ es la codificación de la máquina que rechaza todo. Probemos que esto es una reducción de L_1 a L_2.

En primer lugar, tenemos que f es claramente computable, ya que solo consiste en verificar el formato del input y concatenar con la codificación de una máquina.

En segundo lugar, sea $w \in L_1$, entonces $w = (\mathcal{M})$, por lo tanto $f(w) = (\mathcal{M}, \mathcal{M}_\varnothing)$. Notemos que por la definición de L_1, $L(\mathcal{M}) = \varnothing = L(\mathcal{M}_\varnothing)$, por lo tanto $f(w) \in L_2$.

Finalmente, sea $w \notin L_1$, entonces tenemos dos casos:

1. $w \neq (\mathcal{M})$: En este caso $f(w) = 0 \neq (\mathcal{M}_1, \mathcal{M}_2)$, por lo tanto $f(w) \notin L_2$.
2. $w = (\mathcal{M})$ y $L(\mathcal{M}) \neq \varnothing$: En este caso tenemos que $L(\mathcal{M}) \neq L(\mathcal{M}_\varnothing)$, por lo tanto $f(w) = (\mathcal{M}, \mathcal{M}_\varnothing) \notin L_2$

Por lo tanto $w \in L_1 \Leftrightarrow f(w) \in L_2$, es decir, f es una reducción y por teorema visto en clases tenemos como L_1 es indecidible, L_2 debe ser indecidible.

Parte (a): 3 pts. uno por definir bien $\mathfrak{X}$ es (i) y dos por justificar bien en (ii), si en (ii) solo justifica que la máquina acepta H, pero no que rechaza los inputs fuera de H, dar un punto.
Parte (b): 3 pts. uno por definir bien la reducción y dos por justificar correctamente que es reducción (dar 2.5 pts. si hace todo bien pero olvida mencionar que la función es computable).

Pregunta 2: Semántica en $\mathcal{L}(P)$ y equivalencia lógica

Decida si las siguientes afirmaciones son verdaderas o falsas. Demuestre su respuesta.

(a) Si $L_1 \cup L_2$ es decidible, entonces L_1 y L_2 son decidibles.

(b) Sea $\mathcal{M}_1$ una MT. Si $L_1 = L(\mathcal{M}_1)$, entonces existe una MT $\mathcal{M}_2$ tal que $L(\mathcal{M}_2) = \overline{L_1}$.

Solución

Parte (a)

FALSO. En contraejemplo podría ser $L_1 = \{0,1\}^*$ y $L_2 \subset \{0,1\}^*$ cualquier lenguaje indecidible, por ejemplo el lenguaje diagonal visto en clases $DG = \{w \in \{0,1\}^* : \text{existe } \mathcal{M} \text{ tal que } w = \text{cod}(\mathcal{M}) \text{ y } \mathcal{M} \text{ se detiene en } w\}$. Claramente $L_1 \cup L_2 = L_1$ es decidible, pero L_2 no lo es.

Parte (b)

FALSO. Supongamos que es verdadero. Tomamos $L_1 = DG$, por clases sabemos que existe MT $\mathcal{M}_1$ tal que $L(\mathcal{M}_1) = L_1$, entonces existe $\mathcal{M}_2$ tal que $L(\mathcal{M}_2) = overline L_1$. Por lo tanto, podemos construir una máquina de turing tal que ejecuta $\mathcal{M}_1$ y $\mathcal{M}_2$ en paralelo y acepta el input si $\mathcal{M}_1$ acepta y lo rechaza si $\mathcal{M}_2$ acepta. Esta máquina claramente se detiene con todo input y $L(\mathcal{M}_3) = L_1$, pero sabemos que L_1 es indecidible, por lo que llegamos a una contradicción.

3 pts. por cada parte. En cada una se otorga un punto si se postula que es falso y los otros dos puntos por una correcta justificación.

Pregunta 3: Consecuencia lógica en $\mathcal{L}(P)$

(a) Demuestre que si $L_1 \leq_p L_2$ y $L_2 \leq_p L_3$, entonces $L_1 \leq_p L_3$.

(b) Dada una matriz de enteros A de $n \times m$ y un vector $\vec{b}$ de n enteros, el problema de *programación entera* en su versión de problema de decisión consiste en verificar si existe un vector $\vec{x}$ de m enteros tal que $A\vec{x} \leq \vec{b}$. Por ejemplo, el siguiente es un problema de programación entera para el cual el vector $\vec{x} = (1, 1)$ es adecuado

$$\begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} -1 \\ 0 \\ 3 \end{bmatrix}$$

Como lenguaje, lo definimos por

$$\text{PE} = \{(A, \vec{b}) \mid A \text{ de } m \times n,\ \vec{b} \text{ de } n \times 1 \text{ y existe vector } \vec{x} \text{ de } m \times 1 \text{ tal que } A\vec{x} \leq \vec{b}\}$$

Demuestre que 3SAT $\leq_p$ PE.

Solución

Parte (a)

Supongamos que tenemos lenguajes $L_1 \subset A^*$, $L_2 \subset B^*$ y $L_3 \subset C^*$ tales que $L_1 \leq_p L_2$ y $L_2 \leq_p L_3$, es decir, existen funciones $f : A^* \to B^*$ y $g : B^* \to C^*$ computables tales que

$$w \in L_1 \iff f(w) \in L_2$$
$$w' \in L_2 \iff g(w') \in L_3$$

Ahora notemos que la función $h := g \circ f : A^* \to C^*$ también es computable, ya que simplemente podemos ejecutar una después de la otra y además

$$w \in L_1 \iff f(w) \in L_2 \iff g(f(w)) = h(w) \in L_3$$

Es decir, $L_1 \leq_p L_3$.

Parte (b)

Para probar que 3SAT $\leq_p$ PE, tendremos que construir una función computable $f : \{0,1\}* \to \{0,1\}^*$ tal que

$$w \in \text{3SAT} \Leftrightarrow f(w) \in \text{PE}$$

Claramente, estamos asumiendo que usamos alguna codificación para los elementos de ambos lenguajes.
Para esto, supongamos que $w \in \{0,1\}*$. Si w no es la codificación de una fórmula en 3-CNF, entonces $f(w) = 0$. Si w es la codificación de una fórmula en 3-CNF con n variables y m cláusulas, entonces $f(w)$ será la codificación de $(A, \vec{b})$, donde

1. A es una matriz $(2n+m) \times n$ tal que las primeras n filas son la matriz identidad $I_(n \times n)$, las segundas n filas son la matriz $-I_{n \times n}$ y las últimas m filas son tales que cada una representa una cláusula de la siguiente manera: si la variable i-ésima aparece negada, hay un 1 en la posición i y si aparece normal, hay un -1 en la posición i, el resto se rellena con 0's.

2. $\vec{b}$ es una vector de $2n + m$ variables tales que las primeras n variables son 1's, las segundas n variables son 0's y las últimas m variables son la cantidad de variables que aparecen negadas en cada cláusula (en el mismo orden que en A).

Para que quede claro el funcionamiento de f se presenta un ejemplo, sea w la codificación de

$$\varphi = (w_1 \vee w_2 \vee \neg w_3) \wedge (\neg w_2 \vee \neg w_1 \vee w_4) \wedge (w_2 \vee w_3 \vee w_4)$$

Entonces el sistema representado por $f(w)$ se vería de la siguiente manera,

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & -1 & 1 & 0 \\ 1 & 1 & 0 & -1 \\ 0 & -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 2 \\ 0 \end{bmatrix}$$

Ahora probemos que f es reducción. Claramente es computable, ya que es prácticamente un cambio de formato del input. Además, sea $w \in \{0,1\}^*$, notemos que si w no es codificación de una fórmula 3-CNF, entonces $w \notin$ 3SAT y $f(w) = 0 \notin$ PE.

Ahora supongamos que w es codificación de una fórmula φ 3-CNF. Si tuviéramos una solución x para el sistema relacionado a $f(w)$, entonces por la fórmula que construimos A este debe ser un vector de 0's y 1's, por lo que podríamos asignar a las variables de φ la valuación que hace w_i verdadero si la i-ésima coordenada de x es 1 y la hace falsa si no. Esta valuación satisface φ, por la forma que construimos las últimas filas, ya que si tenemos un literal verdadero en una cláusula, entonces al hacer el producto con las variables, esto será menor que el caso en el que no se cumple ningún literal, es decir, será menor que la suma de variables negadas.

Si w es la codificaciín de una fórmula φ 3-CNF tal que es satisfacible, entonces notemos que el sistema tendrá como solución el vector tal que la i-ésima coordenada es la valuación de la i-ésima variable. Esto se obtiene de forma similar a lo descrito anteriormente.

Por lo tanto, $w \in$ 3SAT $\Leftrightarrow f(w) \in$ PE, es decir 3SAT $\leq_p$ PE.

Si vemos el ejemplo anterior, notemos que la valuación σ que hace todas las variables 1, menos w_2 es una solución para el sistema, por lo tanto tenemos que,

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & -1 & 1 & 0 \\ 1 & 1 & 0 & -1 \\ 0 & -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 2 \\ 0 \end{bmatrix}$$

Parte (a): 1 pt. por enunciar bien las definiciones, 1 pt. por definir h y argumentar por qué es computable y 1 pt. por probar la doble implicancia.
Parte (b): 1 pt. por definir bien la reducción y 2 pts. por argumentar bien por qué (0.5 pts. por computabilidad y 1.5 pts. por la doble implicancia).