

Computabilidad y Complejidad

Tema 5: Incomputabilidad. Problemas decidibles e indecidibles.

Índice:

1. Generalidades.
2. Introducción.
3. Máquinas de Turing normalizadas.
4. Códigos de máquinas de Turing normalizadas.
5. Un lenguaje no recursivamente enumerable: el lenguaje `Diagonal`.
6. El lenguaje `Universal`. La máquina de Turing `Universal`.
7. Algunos ejercicios sobre lenguajes recursivamente enumerables, y no recursivamente enumerables.
8. El teorema de Rice.

Bibliografía Básica Recomendada

- ♦ Introducción a la teoría de autómatas, lenguajes y computación (Hopcroft, John E., Ullman, Jeffrey D., Motwani, Rajeev)
- ♦ Elements of the theory of computation (Lewis, Harry R., Papadimitriou, Christos H.)
- ♦ Teoría de la computación (Brookshear, J. Glenn)

Generalidades

En este tema consideraremos las clases de los lenguajes recursivos, recursivamente enumerables, y no recursivamente enumerables desde el punto de vista de lo qué es o no computable.

Este desarrollo puede trasladarse directamente al estudio de qué problemas de decisión o cuestiones binarias (esto es, del tipo de respuesta: SÍ o NO) son o no algorítmicamente decidibles. En este contexto, las cuestiones a las que nos referimos son aquellas relativas a objetos de un determinado dominio (grafos, gramáticas, autómatas, expresiones, etc.) susceptibles de ser codificados como palabras en un alfabeto dado. Con esta representación a la cuestión puede asociársele el lenguaje de los códigos de los objetos para los que la respuesta es SÍ, y, en consecuencia, cuando la cuestión se aplica a un objeto para obtener la respuesta basta ver si la palabra codificación del objeto pertenece o no al lenguaje asociado a la misma.

Así se tendrá que el lenguaje asociado es:

- ♦ **recursivo** si y sólo si la cuestión es **decidable**,
- ♦ **recursivamente enumerable** pero **no recursivo** si y sólo si la cuestión es **semidecidible**,
- ♦ **no recursivamente enumerable** si y sólo si la cuestión es **indecidable**.

Introducción

En esta sección nos ceñiremos exclusivamente a lenguajes definidos sobre el alfabeto $\{0, 1\}$.

Proposición: Sea un lenguaje recursivamente enumerable sobre el alfabeto $\{0, 1\}$, existe una máquina de Turing con alfabeto de cinta $\{0, 1, B\}$ que lo acepta.

De este modo a partir de aquí podemos suponer, mientras no se diga lo contrario, que todas las máquinas de Turing consideradas son deterministas y tienen como alfabeto de cinta $\{0, 1, B\}$.

Máquinas de Turing normalizadas

Una máquina de Turing con la cinta acotada por la izquierda o no (podemos adoptar para todas las máquinas consideradas una cualquiera de las dos opciones ya que los resultados son idénticos),

$$M = (\Sigma, \Gamma, Q, f, B, q_1, F)$$

diremos que está normalizada siempre que:

- $\Sigma = \{0, 1\}$
- $\Gamma = \{0, 1, B\}$
- $Q = \{q_1, \dots, q_n\}$
- $F = \{q_2\}$

En consecuencia, para definir una máquina de Turing normalizada sólo es necesario definir su función de transición, ya que a partir de la misma queda también definido su conjunto de estados.

Códigos de máquinas de Turing normalizadas

Sea

$$M = (\Sigma, \Gamma, Q, f, B, q_1, F)$$

una máquina de Turing normalizada.

Una transición

$$f(q_i, a_j) = (q_k, a_m, m_n)$$

puede también representarse como

$$q_i | a_j | q_k | a_m | m_n$$

Si se codifica:

- Cada estado q_i como 0^i , $i=1, \dots, \tilde{n}$.

- El símbolo

- 0 como 0
- 1 como 00
- B como 000

- El desplazamiento

- R como 0
- L como 00

y $|$ como 1 puede una transición representarse como

$$0^i 1 0^j 1 0^k 1 0^m 1 0^n$$

donde

$$\blacklozenge j, m \in \{1, 2, 3\}$$

$$\blacklozenge n \in \{1, 2\}$$

El conjunto de transiciones $t_s, s = 1, \dots, r$, de una máquina de Turing normalizada M puede representarse mediante la palabra $x \in \{0, 1\}^*$

$$x = t_1 11 \dots 11 t_r 111$$

A la palabra x se le denomina un código válido de la máquina M .

Observación: Si una máquina de Turing normalizada M tiene r transiciones, entonces tiene $r!$ códigos válidos.

Observación: En una máquina de Turing normalizada consideraremos que cada estado aparece en al menos una de sus transiciones (con excepción del estado inicial).

Observación: En lo que sigue consideraremos cada $x \in \{0,1\}^*$ como un código de máquina de Turing normalizada bien válido bien inválido.

Sea COD el lenguaje de códigos válidos, esto es,

$$COD = \{x \in \{0,1\}^* / x \text{ es un código válido de alguna máquina de Turing normalizada}\}$$

Proposición: El lenguaje COD es un lenguaje recursivo.

En este contexto a cada palabra $x \in \{0,1\}^*$ se le puede asociar una máquina de Turing M_x y un lenguaje L_x sobre el alfabeto $\{0,1\}$ del modo que sigue:

M_x es:

$\Rightarrow M$, si x es un código válido de la máquina de Turing normalizada M ,

\Rightarrow un máquina inválida, totalmente bloqueada, incapaz, por tanto, de realizar transiciones; y que, en consecuencia, reconoce el lenguaje vacío.

y

$L_x = L(M_x)$, esto es:

$\Rightarrow L(M)$, si x es un código válido de la máquina de Turing normalizada M ,

$\Rightarrow \emptyset$, en otro caso.

Un lenguaje no recursivamente enumerable: el lenguaje Diagonal

Definimos el lenguaje al que denominaremos `Diagonal` y que denotaremos por `DIA` como

$$\text{DIA} = \{x \in \{0,1\}^* / x \notin L_x\}$$

Proposición: El lenguaje `DIA` no es recursivamente enumerable.

El lenguaje Universal. La máquina de Turing Universal

Una máquina de Turing universal, denotada por MTU , reconoce un lenguaje que seguidamente definiremos y que denominaremos lenguaje Universal y que denotaremos mediante UNI .

Como pasos necesarios para la definición de UNI realizaremos algunas definiciones previas.

Sean las aplicaciones

$$c: \{0,1\}^* \longrightarrow \{0,1\}^*$$

$$d: \{0,1\}^* \longrightarrow \{0,1\}^*$$

de modo que para cada $x \in \{0,1\}^*$:

- $x = c(x)d(x)$
- $c(x) =$
 - ♦ u , si $u \in \text{COD}$ y $\exists z: x = uz$
 - ♦ x , en otro caso

Definimos ahora el lenguaje `Universal` como:

$$\text{UNI} = \{x \in \{0,1\}^* / d(x) \in L_C(x)\}$$

Propiedad: Dada una máquina de Turing normalizada M y una palabra $x \in \{0,1\}^*$:

$x \in L(M) \Leftrightarrow c_M x \in \text{UNI}$, donde c_M es cualquier código válido de M .

Proposición: El lenguaje `UNI` es recursivamente enumerable ya que $L(\text{MTU}) = \text{UNI}$.

Seguidamente pasamos a definir la `MTU`.

Inicialmente definiremos una máquina con tres cintas que denominaremos MTU' , y cuyo objetivo para una entrada x es, si $c(x) \in COD$, emular a la máquina codificada en $c(x)$ con la entrada $d(x)$. Las cintas de esta máquina reciben los nombres de: entrada, emulación y estado.

Esta máquina opera como sigue: En la cinta de entrada se escribe la palabra x que la máquina debe analizar. En primer lugar la máquina calcula $c(x)$; si $c(x) \notin COD$ termina deteniéndose y rechazando, en otro caso calcula $d(x)$, la copia en la cinta de emulación y coloca su cabezal al principio de la palabra copiada, y escribe 0 en la cinta de estado.

En la cinta de estado en cada momento aparece escrita una palabra de la forma 0^i que representa el estado q_i en el que la máquina codificada en $c(x)$ se encuentra en cada etapa de la emulación.

La emulación se realiza por pasos de la forma que sigue: a partir del símbolo leído por el cabezal de la cinta de emulación y del estado codificado en la cinta de estado busca la transición asociada en $c(x)$ –sobre la cinta de entrada–; si no la encuentra termina rechazando, en otro caso la lleva a cabo: 1) modificando el contenido de la cinta de emulación escribiendo el símbolo especificado y desplazando el cabezal, y 2) actualizando el estado en la cinta de estado; seguidamente comprueba si este estado es final en cuyo caso termina aceptando, en otro caso repite este paso.

En consecuencia, en el caso en que $c(x) \in \text{COD}$ la MTU' emula la computación realizada por la máquina codificada en $c(x)$ con la entrada $d(x)$; así

$$L(\text{MTU}') = \text{UNI}.$$

Una MTU es la MTU' transformada en una máquina equivalente normalizada.

Observación: Nótese que, en consecuencia, una MTU también tendrá sus correspondientes códigos.

Sea COMDIA el lenguaje complementario de DIA , esto es,

$$\text{COMDIA} = \{x \in \{0,1\}^* / x \in L_x\}$$

Proposición: El lenguaje COMDIA es recursivamente enumerable pero no es recursivo.

Proposición: El lenguaje UNI no es recursivo.

Algunos ejercicios sobre lenguajes recursivamente enumerables, y no recursivamente enumerables

Ejercicio: Sea el lenguaje

$$\text{PAR} = \{x \in \{0,1\}^* / c(x) \in \text{COD} \wedge \text{la máquina codificada en } c(x) \text{ se detiene con la entrada } d(x)\}$$

Demuestre que:

- ♦ PAR es recursivamente enumerable.
- ♦ PAR no es recursivo

Ejercicio: Sean los lenguajes:

$$L_v = \{x \in \{0,1\}^* / L_x = \emptyset\} \text{ y } L_{nv} = \{x \in \{0,1\}^* / L_x \neq \emptyset\}$$

Demuestre que:

- L_{nv} es recursivamente enumerable.
- L_v no es recursivamente enumerable.

El teorema de Rice

El teorema de Rice permite contestar de un modo sistemático y directo a la pregunta de si una determinada cuestión sobre el lenguaje reconocido por una máquina de Turing es o no decidable.

Así, en lo que sigue trataremos sobre cuestiones del tipo: “para la máquina de Turing M ¿cumple $L(M)$ la propiedad P ?”. Donde, por ejemplo, la propiedad P podría referirse a la finitud del lenguaje.

Seguidamente introduciremos la noción de propiedad de los lenguajes recursivamente enumerables sobre el alfabeto $\{0,1\}$. Y más específicamente la noción de propiedad no trivial.

Sea $\mathcal{L}_{\text{REN}}(\{0,1\}^*) = \{L \subseteq \{0,1\}^* / L \in \mathcal{L}_{\text{REN}}\}$.

Dada una propiedad P sobre lenguajes $L \subseteq \{0,1\}^*$, definimos

$$\mathcal{P} = \{L \in \mathcal{L}_{\text{REN}}(\{0,1\}^*) / L \text{ cumple } P\}$$

ahora la cuestión inicial da lugar al lenguaje

$$L_{\mathcal{P}} = \{x \in \{0,1\}^* / L_x \in \mathcal{P}\}$$

Si x es un código válido de la máquina de Turing M , se tiene que

“para la máquina de Turing M ¿cumple $L(M)$ la propiedad P ?”

\Leftrightarrow

$$x \in L_{\mathcal{P}}$$

Diremos que una propiedad P de los lenguajes recursivamente enumerables es trivial si

$$\mathcal{P} = \emptyset \quad \vee \quad \mathcal{P} = \mathcal{L}_{\text{REN}}(\{0,1\}^*)$$

en otro caso diremos que es no trivial.

Teorema de Rice: Cualquier propiedad P no trivial de los lenguajes recursivamente enumerables no es decidable, esto es, $\mathbb{L}_{\mathcal{P}}$ no es recursivo.

Ejemplos: No son decidibles las siguientes cuestiones:

- La máquina de Turing M reconoce un lenguaje `finito`.
- La máquina de Turing M reconoce un lenguaje `infinito`.
- La máquina de Turing M reconoce un lenguaje `regular`.