Modelos de Computación y de Complejidad

Grado en Ingeniería Informática. Tecnologías Informáticas (Curso 2021–2022)

ETS Ingeniería Informática - Universidad de Sevilla

RECURSIVIDAD ENUMERABLE E INDECIDIBILIDAD

Ejercicio 1.– Probar que es GOTO–computable la función parcial $f : \mathbb{N} - \to \mathbb{N}$ definida por:

$$f(x) = \begin{cases} 1 & \text{si } \varphi_x(x) \downarrow \\ \uparrow & \text{si } \varphi_x(x) \uparrow \end{cases}$$

Indicación: Pruébese que la gráfica de f (es decir, el conjunto $\{(x, y) \in \mathbb{N} \times \mathbb{N} \mid \mathtt{HALT}(x, x) \wedge y = 1\}$) es recursivamente enumerable (y aplíquese la proposición 5).

Ejercicio 2.– Probar que **no** es GOTO–computable la función total $f : \mathbb{N} \to \mathbb{N}$ definida por:

$$f(x) = \begin{cases} \varphi_x(x) + 1 & \text{si } \varphi_x(x) \downarrow \\ 0 & \text{en otro caso} \end{cases}$$

Indicación: Pruébese el resultado por reducción al absurdo. Para ello, suponiendo que f fuese GOTO–computable, considérese un número natural n tal que $\varphi_n^{(1)} = f$. A partir de este supuesto, llegar a una contradicción.

Ejercicio 3.– Probar que existen funciones tales que:

(a) **No** son GOTO-computables y, en cambio, su **dominio** y su **rango** **sí** son conjuntos GOTO–computables.

(b) **Sí** son GOTO-computables y, en cambio, su **dominio** y su **rango** **no** son conjuntos GOTO–computables.

Ejercicio 4.– Demostrar, por reducción al absurdo, que es vacío el conjunto

$$A = \{n \in \mathbb{N} : \varphi_n^{(1)} \text{ es total y } \forall x \in \mathbb{N}\ (x \in \mathcal{K} \iff \varphi_n^{(1)}(x) = 0)\}$$

Indicación: Pruébese que si existiera un número natural n tal que $n \in A$, entonces el conjunto $\mathcal{K} = \{x \in \mathbb{N} \mid \varphi_x^{(1)}(x) \downarrow\}$ del problema de la parada, sería GOTO-computable.

Ejercicio 5.– Probar que el conjunto

$$A = \{\langle x, y \rangle \in \mathbb{N} : \text{el programa de código } x \text{ para sobre } y \text{ en, \textbf{exactamente}, un número impar de pasos}\}$$

es recursivamente enumerable.

Ejercicio 6.– Sean A y B suconjuntos de $\mathbb{N}^k$, con $k \geq 1$, tales que A es GOTO-computable y $A \cup B$ es **r.e.** Demostrar que el conjunto B ha de ser **r.e.** ¿Puede asegurarse que el conjunto B sea **r.e.** en el caso en que A sea **r.e.** y $A \cup B$ sea GOTO-computable?

Ejercicio 7.– Justificar la veracidad o falsedad de las siguientes afirmaciones.

1. Existen conjuntos de números naturales que ni son `GOTO`-computables ni son recursivamente enumerables.
2. La negación de un predicado `GOTO`-computable es, también, un predicado `GOTO`-computable.
3. La negación de un predicado *parcialmente decidible* es, también, un predicado *parcialmente decidible*.
4. La cuantificación existencial de un predicado `GOTO`-computable es, también, un predicado `GOTO`-computable.
5. La cuantificación existencial de un predicado *parcialmente decidible* es, también, un predicado *parcialmente decidible*.
6. La cuantificación universal de un predicado `GOTO`-computable es, también, un predicado `GOTO`-computable.
7. La cuantificación universal de un predicado *parcialmente decidible* es, también, un predicado *parcialmente decidible*.
8. El dominio y la gráfica de una función `GOTO`-computable (**parcial** o **total**) son conjuntos recursivamente enumerables.
9. El conjunto $A = \{x : \text{ el programa de código } x \text{ para sobre } 2x\}$ es recursivamente enumerable.
10. El teorema de Rice permite demostrar que **ciertos** subconjuntos de $\mathbb{N}$ **no** son `GOTO`–computables.
11. El conjunto de índices de **una** función `GOTO`–computable de aridad 1, es un conjunto `GOTO`–computable.
12. El conjunto de índices de **la** función identidad en $\mathbb{N}$ es un conjunto `GOTO`– computable.

Ejercicio 8.– Aplicando el teorema de Rice, probar que **no** son `GOTO`–computables los conjuntos:

1. $A = \{x \in \mathbb{N} : \text{ dom}(\varphi_x^{(1)}) \text{ es un conjunto finito}\}$.
2. $B = \{x \in \mathbb{N} : \ \varphi_x^{(1)} \text{ es un predicado}\}$.
3. $C = \{x \in \mathbb{N} \ : \ \varphi_x^{(1)} \text{ es una aplicación sobreyectiva}\}$.
4. $D = \{x \in \mathbb{N} : \ \varphi_x^{(1)} \text{ es total y } \forall y\ (\varphi_x^{(1)}(y) \leq \varphi_x^{(1)}(y+1))\}$.
5. $E = \{x \in \mathbb{N} : \forall x \in \mathbb{N}\ (\varphi_x^{(1)}(y) \text{ es una potencia de } 2)\}$.
6. $F = \{x \in \mathbb{N} : \text{ el programa de código } x \text{ calcula la función identidad en } \mathbb{N} \text{ o bien la aplicación vacía}\}$.
7. $G = \{x \in \mathbb{N} : \text{ dom}(\varphi_x^{(1)}) \ \underline{\text{no}} \text{ es GOTO-computable})\}$.
8. $H = \{x \in \mathbb{N} : \text{ rang}(\varphi_x^{(1)}) \ \underline{\text{no}} \text{ es GOTO-computable})\}$.

Indicación: Para probar que un cierto conjunto $A \subseteq \mathbb{N}$ **no** es `GOTO`-computable, aplicando el teorema de Rice, hay que considerar un conjunto $\mathcal{F}$ de funciones `GOTO`-computables 1-aria, cuyo conjunto de índices $I_{\mathcal{F}} = \{n \in \mathbb{N} \mid \varphi_n^{(1)} \in \mathcal{F}\}$, coincida con A y, de tal manera que $\mathcal{F}$ sea no vacío y distinto de $\texttt{GCOMP}^{(1)}$.

Ejercicio 9.– Se considera el siguiente problema de decisión:

"Dado un número natural $n \in \mathbb{N}$, determinar si $n \in rang(\varphi_n^{(1)})$"

Probar que dicho problema es indecidible.

Indicación: Pruébese, por reducción al absurdo, que el conjunto $A = \{n \in \mathbb{N} : \ n \in rang(\varphi_n^{(1)})\}$ **no** es `GOTO`–computable. Si lo fuese, entonces sería `GOTO`–computable la función parcial $f : \mathbb{N} - \to \mathbb{N}$ definida por $f(x) = x$, si y sólo si $x \notin A$. Luego, existiría $e \in \mathbb{N}$ tal que $f = \varphi_e^{(1)}$. A partir la existencia de un tal número natural e, llegar a una contradicción.

Ejercicio 10.– Se considera el siguiente problema de decisión:

"Dado un número natural $n \in \mathbb{N}$, determinar si el dominio de la función $\varphi_n^{(1)}$ es un conjunto `GOTO`*-computable"*

Probar que dicho problema es indecidible.

Indicación: Considérese el conjunto $A = \{n \in \mathbb{N} \mid dom(\varphi_n^{(1)})$ es un conjunto `GOTO`-computable$\}$ y aplíquese el teorema de Rice considerando el conjunto de funciones

$$\mathcal{F} = \{f \in \texttt{GCOMP}^{(1)} \mid dom(f) \text{ es un conjunto } \texttt{GOTO}\text{-computable}\}$$

(para probar que $\mathcal{F} \neq \emptyset$, téngase presente que en el ejercicio 1 se proporciona una función cuyo dominio **no** es un conjunto `GOTO`-computable).

Ejercicio 11.– Se considera el siguiente problema de decisión:

"Dado un número natural $n \in \mathbb{N}$, determinar si la gráfica de la función $\varphi_n^{(1)}$ **no** *es un conjunto* `GOTO`*-computable"*

Probar que dicho problema es indecidible.

Ejercicio 12.– Se considera el siguiente problema de decisión:

"Dado un número natural $n \in \mathbb{N}$, determinar si existe un programa `GOTO` *sin instrucciones condicionales que calcula la función $\varphi_n^{(1)}$"*

Probar que dicho problema es indecidible.

Indicación: Considérese el conjunto $A = \{n \in \mathbb{N} \mid$ Existe un programa `GOTO` sin condicionales que calcula $\varphi_n^{(1)}\}$ y aplíquese el teorema de Rice considerando el conjunto de funciones

$$\mathcal{F} = \{f \in \texttt{GCOMP}^{(1)} \mid \text{ Existe un programa } \texttt{GOTO} \text{ sin condicionales que calcula } f\}$$

(para probar que $\mathcal{F} \neq \texttt{GCOMP}^{(1)}$ recuérdese que las funciones de aridad 1 que calcula los programas `GOTO` sin condicionales son, exactamente, las funciones constantes).

Ejercicio 13.– Consideremos el siguiente problema de decisión:

"Dado un número natural n, determinar si la computación del programa `GOTO` *de código n, con dato de entrada n,* **no es de parada***"*

Probar que dicho problema **no es** *semidecidible* y, por tanto, es indecidible.

Indicación: Para probar que dicho problema **no es** *semidecidible* se ha de demostrar que el conjunto $A = \{n \in \mathbb{N} \mid \varphi_n^{(1)}(n) \uparrow)\}$, asociado a dicho problema, **no es r.e.** Para ello, obsérvese que su complementario $\overline{A}$ coincide con el conjunto $\mathcal{K}$ del problema de la parada; es decir, $\overline{A}$ es un conjunto **r.e.** pero **no** es `GOTO`-computable. De ahí, dedúzcase que el conjunto A no puede ser **r.e.** ya que, si lo fuera, entonces aplicando el *teorema del complemento* resultaría que A sería `GOTO`-computable y, por tanto, también lo sería $\overline{A}$, llegando a una contradicción.

Ejercicio 14.– Consideremos los siguientes problemas de decisión:

$X \equiv$ *"Dado un número natural n, determinar si la función $\varphi_n^{(1)}$* **no es** *inyectiva"*
$Y \equiv$ *"Dado un número natural n, determinar si la función $\varphi_n^{(1)}$* **sí es** *inyectiva"*

Demostrar que el problema X es **indecidible** pero que, en cambio, **sí es semidecidible**. Demostrar que el problema Y es **indecidible** y que, además, **no es semidecidible**.

e, incluso, **no es** *semidecidible.*

Indicación: Para probar que los problemas X e Y son indecidibles, basta probar que los conjuntos $A_X = \{n \in \mathbb{N} \mid \varphi_n^{(1)}$ **no** es inyectiva$\}$ y $A_Y = \{n \in \mathbb{N} \mid \varphi_n^{(1)}$ es inyectiva$\})$ **no** son GOTO-computables. Para ello, basta aplicar el teorema de Rice al conjunto $\mathcal{F} = \{f \in \mathtt{GCOMP}^{(1)} \mid f$ **no** es inyectiva$\}$ para A_X, y al conjunto $\mathcal{F} = \{f \in \mathtt{GCOMP}^{(1)} \mid f$ es inyectiva$\}$ para A_Y.

Para probar que el problema X es semidecidible, basta demostrar que el conjunto A_X es **r.e.**; es decir, que la correspondiente relación de pertenencia es un predicado parcialmente decidible:

$$n \in A_X \iff \varphi_n^{(1)} \text{ **no** es inyectiva} \iff [\exists z \exists t \exists u\ (z \neq t \wedge (z,u) \in G(\varphi_n^{(1)}) \wedge (t,u) \in G(\varphi_n^{(1)}))].$$

Finalmente, para probar que el problema Y **no es** semidecidible, basta demostrar que el conjunto A_Y no es **r.e.** (si lo fuera, teniendo presente que A_Y es el complementario A_X y que A_X es r.e, del *teorema del complemento* se deduciría que el conjunto A_X sería. GOTO-computable).

Ejercicio 15.– Consideremos el siguiente problema de decisión:

"Dado un número natural n, determinar si el rango de la función $\varphi_n^{(1)}$ es un conjunto de números primos"

Demostrar que el problema X es indecidible y que, además, **no es** *semidecidible.*

Indicación: Para probar que el problema propuesto es indecidible, basta demostrar que **no** es GOTO-computable el conjunto $A = \{n \in \mathbb{N} \mid \mathrm{rang}(\varphi_n^{(1)}) \subseteq \mathtt{PRIMO})\}$, en donde PRIMO es el conjunto de los números naturales que son primos. Para ello, basta aplicar el teorema de Rice al conjunto $\mathcal{F} = \{f \in \mathtt{GCOMP}^{(1)} \mid \mathrm{rang}(f) \subseteq \mathtt{PRIMO})\}$.

Para probar que el problema propuesto **no** es semidecidible, se ha de demostrar que A **no** es **r.e.** Para ello, se prueba que el conjunto complementario $\overline{A}$ es **r.e.** y, por tanto, si el conjunto A fuese **r.e.** del del *teorema del complemento* se deduciría que el conjunto A sería GOTO-computable. Lo que es una contradicción. Para probar que $\overline{A}$ es **r.e.** téngase presente que:

$$n \in A \iff \exists x \exists t\ (\mathtt{STEP}^{(1)}(x,n,t) \wedge \forall z < t\ (\neg \mathtt{STEP}^{(1)}(x,n,z)) \wedge (r(d.i.^{(1)}(x,n,t)))_1 \text{ no es un número primo}$$

Ejercicio 16.– Consideremos el siguiente problema de decisión:

"Dado un número natural n, determinar si para cualquier número natural $x \in \mathbb{N}$ se tiene que $\varphi_n^{(1)}(x)$ es un número par"

Demostrar que el problema X es indecidible y que, además, **no es** *semidecidible.*

Indicación: Para probar que el problema propuesto es indecidible, basta demostrar que el conjunto $A = \{n \in \mathbb{N} \mid \forall x\ (\varphi_n^{(1)}(x)$ es un número par$)\}$ **no** es GOTO-computable. Para ello, basta aplicar el teorema de Rice al conjunto $\mathcal{F} = \{f \in \mathtt{GCOMP}^{(1)} \mid \forall x\ (f(x)$ es un número par$)\}$.

Para probar que el problema propuesto **no** es semidecidible, se ha de demostrar que A **no** es **r.e.** Para ello, se prueba que el conjunto complementario $\overline{A}$ es **r.e.** y, por tanto, si el conjunto A fuese **r.e.** del *teorema del complemento* se deduciría que el conjunto A sería GOTO-computable. Lo que es una contradicción.