El nivel 0 de la Jerarquía de Grzegorzcyk es recursivamente enumerable

Joaquín Díaz Boils
Pontificia Universidad Católica del Ecuador
boils@uji.es

José P. Úbeda Rives
Universitat de València
Jose.P.Ubeda@uv.es

Resumen

En la presente comunicación se afronta una descripción de las clases de la Jerarquía de Grzegorzcyk por medio de una caracterización constructiva del operador *recursión* acotada bajo el cual están cerradas dichas clases. Ello permite concluir, a partir de la demostración para el nivel cero, que las clases pertenecientes a dicha jerarquía son todas ellas recursivamente enumerables.

Palabras clave: Funciones recursivas primitivas, Jerarquía de Grzegorzcyk, conjuntos recursivamente enumerables.

Abstract

In this paper we face a description of the classes belonging to the known as Grzegorzcyk Hierarchy by means of a constructive characterization of the *bounded recursion* operator under which they are closed. That will allow us to conclude that, from the proof at the zero level, all classes in the Grzegorzcyk Hierarchy are recursively enumerable.

Key words: Primitive recursive functions, Grzegorzcyk Hierarchy, recursively enumerable sets.

Introducción

La importancia de la Jerarquía de Grzegorczyk, dentro del campo de la Teoría de la Computabilidad, reside en el hecho de que permite una clasificación de las funciones recursivas primitivas según su complejidad. Esto es, una función es recursiva primitiva si y sólo si existe una clase dentro de la Jerarquía de Grzegorczyk que la contiene y cuyo índice denota el número de *recursiones anidadas* usadas en su definición formal.

De las diferentes caracterizaciones de las clases de la Jerarquía de Grzegorczyk se ha seleccionado para el presente trabajo aquélla en la que se utiliza el operador Recursión acotada. Justamente la dificultad en una posible enumeración de las funciones de las clases citadas reside en dicho operador, dado que involucra a tres funciones a diferencia del operador de recursión usual que sólo supone considerar dos funciones.

El presente estudio ofrece, en nuestra opinión, una descripción más sencilla que la desarrollada en [Axt] y es por tanto adecuada para obtener una enumeración de la clase de las funciones recursivas primitivas.

Definiciones básicas

Andrzej Grzegorczyk en [Grz. páginas 28-29] define una sucesión o jerarquía $\mathcal{E}^n$ de clases de funciones numéricas.[1] Esta sucesión cumple dos propiedades importantes:

- Para cada $n \in \mathbf{N}$, $\mathcal{E}^n$ está incluida propiamente en $\mathcal{E}^{n+1}$.
- La unión de toda la sucesión, es decir, $\bigcup_{n\in\mathbf{N}} \mathcal{E}^n$, es la clase de todas las funciones recursivas primitivas.

Se sabe que la clase de las funciones parciales recursivas y de las funciones recursivas primitivas son recursivamente enumerables (en adelante RE).[2]

[1] Funciones cuyo dominio es $\mathbf{N}^K$ para $k \in \mathbf{N} - \{0\}$ y su rango está incluido en $\mathbf{N}$.
[2] Un conjunto o clase A es RE si y sólo si existe una función computable cuyo rango

En este trabajo mostramos que cada una de las clases $\mathcal{E}^n$ es RE. En concreto lo mostramos para la clase $\mathcal{E}^0$. Pudiendo deducirse entonces fácilmente que las restantes clases de la jerarquía o sucesión de Grzegorczyk también lo son.

La clase $\mathcal{E}^0$ es la clase más pequeña tal que:

1. incluye las funciones

   a) *sucesor*, denotada por s, tal que $s(x)=x+1$;
   b) *primera proyección*, denotada por U_1, tal que $U_1(x,y)=x$;
   c) *segunda proyección*, denotada por U_2, tal que $U_2(x,y)=y$;
   d) $f_0(x,y)=y+1$;[3]

2. y está cerrada respecto a las operaciones

   a) *sustitución*: de las funciones $f(x_1, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots, x_n)$ y $g(y_1, \ldots, y_m)$ se obtiene al sustituir la variable k en f por la función g la función

   $$h(x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n, y_1, \ldots, y_m)=$$

   $$f(x_1, \ldots, x_{k-1}, g(y_1, \ldots, y_m), x_{k+1}, \ldots, x_n);$$

   b) *identificación* de variables: de $f(x_1, \ldots, x_j, \ldots, x_k, \ldots, x_n)$ al identificar las variables x_j y x_k se obtiene la función

   $$f(x_1, \ldots, y, \ldots, y, \ldots, x_n)$$

   c) *sustitución por* la constante *cero*:[4] de $f(x_1, \ldots, x_j, \ldots, x_n)$ al sustituir la variable x_j por 0 se obtiene la función

   $$f(x_1, \ldots, 0, \ldots, x_n)$$

   d) *recursión acotada*:[5] de las funciones $g(\bar{a})$, $h(\bar{a},x,y)$ y $j(\bar{a},x)$ se obtiene la función $f(\bar{a},x)$ por recursión de g y h acotada por j si se cumplen las condiciones

   1) $f(\bar{a},0)=g(\bar{a})$

   2) $f(\bar{a},s(x))=h(\bar{a},x,f(\bar{a},x))$

   3) $f(\bar{a},x)\leq j(\bar{a},x)$.

---

es A. Es decir, existe un algoritmo que lista todos los elementos de A.

[3] Esta función puede eliminarse de la definición, ya que esta se obtiene a partir de U_2 y s por medio de la sustitución.

[4] Aplicando esta operación podemos tener funciones con cero argumentos. Así que consideramos que dicha operación se aplica sólo a funciones que tengan dos o más argumentos.

[5] Usaremos $\bar{a}$ para denotar $a_1, \ldots, a_n$ para una n dada.

Para definir las clases $\mathcal{E}^n$ se utiliza una serie de funciones auxiliares E_n dadas por:

$$E_0(x,y)=x+y$$

$$E_1(x)=x^2+2$$

$$E_n(x)= E^x_{n-1}(2) \quad \text{si } 1\leq n$$

donde E^x_{n-1} expresa la iteración x-ésima de la función E_{n-1}. Con estas funciones definimos $\mathcal{E}^n$ como la menor de las clases de funciones tales que contienen a las funciones:

- E_k para k<n, constantemente cero, sucesor y proyecciones π_n^m (ver el HECHO 1 más abajo)

y están cerradas bajo:

- composición *C* (ver el HECHO 3 más abajo) y recursión acotada.

Enumeración de la clase $\mathcal{E}^0$

A partir de esta definición no es fácil establecer que $\mathcal{E}^0$ es RE ya que la recursión acotada no es un operador que aplicado a tres funciones g, h y j (con n, n+2 y n+1 argumentos respectivamente) nos dé una función con n+1 argumentos.[6]

Sin embargo, es fácil establecer los hechos siguientes:

<u>HECHO 1</u>. *A partir de* U_1 *y* U_2 *por medio de las operaciones de sustitución e identificación de variables pueden definirse las funciones de proyección* π_n^m*, donde* $0 < n \leq m$*, con* m *variables tal que*

$$\pi_n^m(x_1, \ldots, x_m)=x_n.$$

a) π_1^1 *se obtiene de* U_1 *al identificar sus dos variables.*
b) π_1^2 *es* U_1 *y* π_2^2 *es* U_2.
c) *Supongamos que* π_n^m *con* $0 < n \leq m$ *están definidas, entonces* π_k^{m+1} *se define por casos:*

1. $\pi_k^{m+1}(x_1, \ldots, x_m, x_{m+1})=U_1(\pi_k^m(x_1, \ldots, x_m), x_{m+1})$ *si* $k \leq m$

[6] No existe un algoritmo para decidir si dadas tres funciones g, h y j, la f definida por las condiciones 2d1 y 2d2 cumple la condición 2d3.

2. $\pi_k^{m+1}(x_1, \ldots, x_m, x_{m+1}) = U_2(\pi_k^m(x_1, \ldots, x_m), x_{m+1})$
si $k = m + 1$.

<u>HECHO 2</u>. *Las siguientes funciones pertenecen a* $\mathcal{e}^0$

- *La función cero,* z*, con un argumento, tal que* $z(x) = 0$.[7]
- *la función* $\chi(x, y)$ *tal que*[8]

$$\chi(x, y) = \begin{cases} 1 & \textit{si } x < y, \\ 0 & \textit{en otro caso} \end{cases}$$

- *la función* $\sigma(x, y)$ *tal que*[9]

$$\sigma(x, y) = \begin{cases} x & \textit{si } y = 0, \\ 0 & \textit{en otro caso.} \end{cases}$$

<u>HECHO 3</u>. *Si se tienen las funciones de proyección* π_n^m *y la función* z*, las operaciones de sustitución (de funciones y por la constante cero) e identificación de variables pueden definirse a partir de la operación composición,* C*, que se aplica a un* $n + 1$*-tuplo de funciones* $(f, g_1, \ldots, g_n)$ *tal que*

- f *tiene* n *argumentos*
- *y, para un* $m \in \mathbf{N}^+$*, cada* g_i *(*$i = 1, \ldots, n$*) tiene* m *argumentos*

y nos da una función $C(f, g_1, \ldots, g_n)$ *de* m *argumentos tal que*

$C((f, g_1, \ldots, g_n))(x_1, \ldots, x_m) = f(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))$.

<u>Demostración</u>:

La sustitución se obtiene por medio de la composición

$$C(f, \pi_1^m, \ldots, \pi_{k-1}^m, g, \pi_{k+1}^m, \ldots, \pi_n^m)$$

mientras que la identificación de variables por medio de

$$C(f, \pi_1^m, \ldots, \pi_i^m, \ldots, \pi_i^m, \ldots, \pi_n^m)$$

---

[7] $z(x) = U_1(0, x) = U_2(x, 0)$.
[8] Véase [Grz. página 30].
[9] $\sigma(x, 0) = U_1(x, x)$; $\sigma(x, s(y)) = z(\pi_3^3(x, y, \sigma(x, y)))$; $\sigma(x, y) \leq U_1(x, y)$.

HECHO 4. *Sea* R *un operador que aplicado a tres funciones* g, h *y* j *(con* n, n+ 2 *y* n + 1 *argumentos respectivamente) define una función* f = R(g, h, j) *con* n+1 *argumentos tal que*

- $f(\bar{a},0) = \sigma(g(\bar{a}),\chi(j(\bar{a},0),g(\bar{a})))$
- $f(\bar{a},s(y)) = \sigma(h(\bar{a},x,f(\bar{a},x)),\chi(j(\bar{a},s(y)),h(\bar{a},x,f(\bar{a}, x))))$.

la función R *tiene las siguientes propiedades.*

1. *Si* g, h *y* j *pertenecen a* $\mathcal{e}^0$, R(g, h, j) *también pertenece a* $\mathcal{e}^0$.[10]
2. R(g, h, j)(ā, x) ≤ j(ā, x).
3. *Si* t *es una función definida por recursión acotada a partir de las funciones* g, h *y* j, *entonces* t *es igual a* R(g, h, j).[11]

<u>Demostración</u>:

1. La condición 1. Se satisface dado que se trata de una composición de funciones pertenecientes a $\mathcal{e}^0$ aplicando el HECHO 2.
2. La condición 2. se prueba del modo siguiente:

$$f(\bar{a},0) = \sigma(g(\bar{a}),\chi(j(\bar{a},0),g(\bar{a})))$$

$$= \begin{cases} g(\bar{a}) & \textit{si}\ \ \chi(j(\bar{a},0),g(\bar{a})) = 0 \\ 0 & \textit{en otro caso.} \end{cases}$$

$$= \begin{cases} g(\bar{a}) & \textit{si}\ \ g(\bar{a}) \leq j(\bar{a}, 0) \\ 0 & \textit{en otro caso.} \end{cases}$$

Para el valor R(g,h,j)(ā,0) y además:

$$f(\bar{a},sx) = \sigma(h(\bar{a},x,f(\bar{a},x)),\chi(j(\bar{a},s(y)),h(\bar{a},x,f(\bar{a},x))))$$

$$= \begin{cases} h(\bar{a},x,f(\bar{a},x)) & \textit{si}\ \ \chi(j(\bar{a},s(y)),h(\bar{a},x,f(\bar{a},x)))= 0 \\ 0 & \textit{en otro caso.} \end{cases}$$

$$= \begin{cases} h(\bar{a},x,f(\bar{a},x)) & \textit{si}\ \ h(\bar{a},x,f(\bar{a},x)))\leq j(\bar{a},s(y)) \end{cases}$$

---

[10] Lo mismo vale sustituyendo $\mathcal{e}^0$ por $\mathcal{e}^n$.

[11] Si g, h y j son funciones con el número adecuado de argumentos R(g, h, j) siempre es una función, pero puede suceder que a partir de g, h y j no se obtenga ninguna función por recursión acotada.

0 *en otro caso.*

Para el valor R(g,h,j) (ā,sx).

A partir de los hechos anteriores es fácil establecer que la clase $\mathcal{E}^0$ es la clase más pequeña que

1. incluye las funciones *sucesor*, *cero*, las funciones de *proyección* π_n^m, donde $0 < n \leq m$, y las funciones σ y χ
2. y está cerrada respecto a los operadores C y R cuando se aplican a las funciones con un número adecuado de argumentos.

Enumeración de $\mathcal{E}^0$

A partir de los hechos y las definiciones presentadas puede construirse un algoritmo, cuya descripción explícita se expone a continuación, tal que liste todas las funciones en $\mathcal{E}^0$, por lo tanto se deduce que $\mathcal{E}^0$ es RE.

Para enumerar las funciones del nivel cero consideramos un alfabeto *A* que consta del conjunto de los 12 símbolos

$$\{z, s, \sigma, \chi, p, *, \#, C, R, (,)\}$$

y de la propia coma. Sabemos que cada función en el nivel $\mathcal{E}^0$ puede expresarse por medio de una palabra en *A*, es decir, perteneciente a A^* y donde expresamos:

- las funciones cero y sucesor por las palabras z y s
- las funciones σ y χ por sí mismas.

Para expresar las funciones π_k^m usamos las expresiones del conjunto *P* definido como sigue:

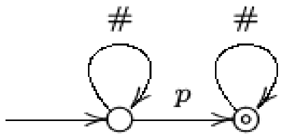
1. p pertenece a *P*: corresponde a π_1^1
2. si $\varphi \in P$, $\varphi\# \in P$: si φ corresponde a π_n^m entonces $\varphi\#$ pertenece a π_n^{m+1}
3. si $\varphi \in P$, $\#\varphi \in P$: si φ corresponde a π_n^m entonces $\#\varphi$ pertenece a π_n^{m+1}

Así, la palabra

$$\# \cdots \# \, p \, \# \cdots \#$$
(n veces)   (m veces)

representa la función π_{n+1}^{m+n+1}.

Un autómata finito que reconoce *P* es



Definimos el conjunto *Pr* de las palabras que representan funciones de e^0 se define inductivamente, junto con el número ρ de argumentos de dicha función:

1. $z \in Pr$ y $\rho(z) = 1$
2. $s \in Pr$ y $\rho(s) = 1$
3. si $\alpha \in P$, entonces $\alpha \in Pr$ y $\rho(\alpha) = |\alpha|$[12]
4. si $\alpha \in Pr$, $\rho(\alpha) = m$ y, para $1 \leq i \leq m$, $g_i \in Pr$ y $\rho(g_i) = n$, entonces

$$C(\alpha, g_1, ..., g_m) \in Pr \text{ y } \rho(C(\alpha, g_1, ..., g_m)) = n$$

5. Si $\alpha \in Pr$, $\rho(\alpha) = m$, $\beta \in Pr$, $\rho(\beta) = m + 2$, $\gamma \in Pr$ $\rho(\gamma) = m + 1$ entonces $R(\alpha, \beta, \gamma) \in Pr$ y $\rho(R(\alpha, \beta)) = m + 1$.

Pr es decidible, es decir, existe un algoritmo que, dada cualquier palabra perteneciente a A^*, responde si, si la expresión pertenece a *Pr* y responde no si no pertenece a *Pr*. Un procedimiento de decisión, dada una palabra m en el lenguaje A^*, es el siguiente:

1. Si m es el símbolo “z” o “s”, pertenece a $A*$ y $\rho(m) = 1$.
2. Si m es el símbolo “σ” o “χ”, pertenece a $A*$ y $\rho(m) = 1$.
3. Si $m \in P$, $m \in Pr$, pertenece a $A*$ y $\rho(m) = t + 1$, donde t es el número de símbolos # que ocurren en m.
4. En otro caso, si los dos primeros símbolos por la izquierda de m son distintos de “C(” y de “R(” o su último símbolo es distinto de “)”, entonces m no pertenece a *Pr*.

[12] Si α es una palabra, denotamos su longitud por |α|. Corresponden a las funciones π_n^m.

5. Si m comienza por "C(" y termina en ")", sea w la expresión obtenida de m al quitar sus dos primeros símbolos y el último símbolo. Para que m ∈ *Pr* tiene que haber s comas fuera de paréntesis la expresión f que va desde el primer símbolo de w hasta la primera coma fuera de paréntesis (sin incluir ésta) debe ser tal que f ∈ *Pr* y ρ(f) = s, además las restantes expresiones que ocurren entre dichas comas (incluida la que está a la derecha de la última coma) deben pertenecer a *Pr* y tener todas ellas la misma aridad t. Si se cumple todo lo dicho, ρ(m) = t y m ∈ *Pr*.

6. Si m comienza por "R(" y termina en ")", sea w la expresión obtenida de m al quitar sus dos primeros símbolos y el último símbolo.

Para que m ∈ *Pr* tiene que haber 2 comas fuera de paréntesis, así w constará de una expresión f que llega hasta la primera coma, otra g que va desde la primera a la segunda coma y otra h que va desde la última coma hasta el final. Además, debe cumplirse que f, g, h ∈ *Pr*, ρ(f) = s, ρ(g) = s + 2 y ρ(h) = s + 1 para un s > 0. Si se cumple todo lo dicho, m ∈ *Pr* y ρ(m) = s + 1.

Conclusión

Se ha logrado enumerar las funciones de la clase $\mathcal{E}^0$ por medio de una reescritura del operador recursión acotada. Se observa asimismo que el procedimiento desarrollado permite su aplicación a las sucesivas clases $\mathcal{E}^k$ con k≥1 y por lo tanto el resultado obtenido puede generalizarse a todas las clases de Grzegorzcyk. Se deduce que todas las clases de la Jerarquía de Grzegorzcyk son recursivamente enumerables.

Referencias:

[Grz.] A. Grzegorczyk. *Some classes of recursive functions*. Rozprawy Matematyczne, 1953.

[Axt] Axt, P. (1963), *Enumeration and the Grzegorczyk Hierarchy.* Mathematical Logic.