Apuntes sobre las Funciones Recursivas Parciales

Programación y Algorítmica Avanzadas – GIA – FIB

José Luis Balcázar – Dept CS – UPC – Revisión de abril de 2024

1 Preliminares

Salvo indicación en contrario, todos los conjuntos son subconjuntos de $\mathbb{N}$. Salvo indicación en contrario, todas las funciones son funciones parciales de $\mathbb{N} \to \mathbb{N}$. Una función parcial será una aplicación de un conjunto de naturales $A \subseteq \mathbb{N}$ en los naturales. El conjunto A es el dominio de la función, y cuando $A = \mathbb{N}$ diremos que se trata de una función total. Representaremos, en general, por letras griegas (como ϕ, ψ, ρ) funciones parciales (que pueden ser totales o no serlo), y reservaremos letras latinas (como f, g, h) para funciones totales. Ocasionalmente usaremos como nombres de función identificadores alusivos a su comportamiento. Veamos algunos ejemplos: la función "sucesor", $\mathtt{suc}(x) = x + 1$; la función "anterior",

$\mathtt{ant}(x) = \begin{cases} 0 & \text{si } x = 0, \\ x - 1 & \text{en otro caso} \end{cases}$; la función diferencia modificada, $x \dot{-} y = \begin{cases} x - y & \text{si } x \geq y \\ 0 & \text{si } x \leq y \end{cases}$;

y la función signo, $\mathtt{sg}(x) = \begin{cases} 0 & \text{si } x = 0 \\ 1 & \text{si } x > 0 \end{cases}$.

La función "anterior" `ant` es una inversa a la función sucesor: habida cuenta de que no trabajamos sobre los enteros, es preciso concretar qué significa $\mathtt{ant}(0)$, como hemos hecho. De la misma manera, es preciso extender la función diferencia, obteniéndose así nuestra diferencia modificada. Puesto que trabajamos con funciones parciales, podríamos haberlas dejado indefinidas en determinados puntos, sin obtener diferencias de calado en la teoría que desarrollaremos; nuestra elección es la que nos resultará más cómoda para este desarrollo.

Insistimos en que, en nuestro contexto, la palabra "parcial" ha de entenderse siempre como "posiblemente parcial": cuando decimos que una función es parcial, significa que no nos pronunciamos sobre su totalidad o no totalidad, dejando abiertas ambas posibilidades; no descartamos que su dominio sea todo $\mathbb{N}$, y así las funciones parciales incluyen las funciones totales.

Denotamos por Dom ϕ el dominio de la función parcial ϕ. Abreviamos $x \in \text{Dom } \phi$ por $\phi(x)\downarrow$, y $x \notin \text{Dom } \phi$ por $\phi(x)\uparrow$. Como veremos, $\phi(x)\uparrow$ significa intuitivamente que el programa que intenta calcular esa función no termina nunca sobre el dato x.

La composición de funciones parciales $(\phi \circ \psi)(x)$, donde ϕ y ψ son funciones parciales, evalúa a $\phi(\psi(x))$, si $\psi(x)$ está definida y además ϕ está definida sobre el resultado, quedando indefinida en otro caso.

Denominaremos "funciones constantes" a las funciones totales que cumplen $f(x) = c$ para alguna $c \in \mathbb{N}$ y para todo x; frecuentemente identificaremos cada constante $c \in \mathbb{N}$ con la correspondiente función constante.

La "función característica" $\chi_A : \mathbb{N} \to \mathbb{N}$ de un conjunto $A \subseteq \mathbb{N}$ es $\chi_A(x) = \begin{cases} 0 & \text{si } x \notin A \\ 1 & \text{si } x \in A \end{cases}$.
Cuando una función total tiene su imagen incluida en $\{0, 1\}$, se trata de una función característica, y podemos identificarla con el correspondiente conjunto. Resulta útil ver las funciones características también bajo un segundo disfraz: predicados, es decir, funciones con valores booleanos, en cuyo caso las operaciones de unión, intersección y complementación se denominan, respectivamente, disyunción, conjunción y negación.

Podemos ver cada función ϕ como un "problema funcional": el de calcular $\phi(x)$, dado x. Cuando ϕ es la función característica de un conjunto A, podemos verla también como un "problema decisional": el de decidir si un valor x dado pertenece o no a A; si pensamos en las dos posibles respuestas booleanas, negativa y positiva, como codificadas por los símbolos 0 y 1, respectivamente, vemos que se trata de una formulación equivalente al correspondiente problema funcional.

2 Codificación de tuplas

Aquí van el juez y el *gangster*
los dos juntos en el mismo verso.

León Felipe: Yo soy el gran blasfemo

La posibilidad de codificar conjuntamente, en un único número natural, un conjunto finito de ellos, e incluso otra información no numérica, fue un descubrimiento sorprendente para los matemáticos de los años treinta del siglo pasado. Es, sin embargo, una obviedad para el informático de hoy, sea profesional o aficionado, que sabe que las grandes bases de datos con cantidades ingentes de información se pueden representar en ficheros sobre un soporte adecuado, frecuentemente magnético, mediante una larga secuencia de *bits*; secuencia que podemos identificar con un (astronómico) número natural, escrito en binario.

Podemos codificar un par de naturales en uno solo mediante la biyección

$$\langle x.y \rangle = (1/2)(x + y)(x + y + 1) + x + 1$$

entre $\mathbb{N} \times \mathbb{N}$ y $\mathbb{N} - \{0\}$. Obsérvese que en $\langle x.y \rangle$ separamos los argumentos por un punto, y no por una coma.

Insistimos en que el cero no codifica ningún par: nos será útil hacerlo así, si bien bastaria retirar el último sumando para obtener una biyección entre $\mathbb{N} \times \mathbb{N}$ y $\mathbb{N}$. El empleo de un punto como separador es un lejano homenaje al "dotted pair" de LISP, cuyas listas corresponderán estructuralmente a nuestras secuencias y cuyo "NIL" corresponderá al cero, que no codifica ningún par.

Extendiendo el concepto de par, definimos una operación entre funciones, que denominamos "formación de pares", como sigue: a partir de dos funciones parciales cualesquiera ϕ, ψ, construimos la función que a cada x le asocia el par $\langle \phi(x).\psi(x) \rangle$. Denotamos esta función $\langle \phi.\psi \rangle : \mathbb{N} \to \mathbb{N}$.

Extendemos el proceso a tres naturales repitiendo la operación:

$$\langle x.y.x \rangle = \langle x.\langle y.z \rangle \rangle$$

Nótese que sobrecargamos el símbolo, dejando que sea el contexto el que indique si se trata de la operación que codifica dos números o de la que codifica tres; o cuatro, quince, o dos mil novecientas veintiocho componentes: para cada $m > 1$, tenemos una operación distinta, de $\mathbb{N}^m \to \mathbb{N}$, que codifica m naturales en uno solo. Nótese que, debido al sumando 1 en la definición del par, estas funciones son inyectivas pero no biyectivas. Para simplificar la notación, al componer estas funciones de codificación con otras funciones escribiremos frecuentemente $\phi(x, y, z)$, por ejemplo, en vez de $\phi(\langle x.y.z\rangle)$; pero ha de quedar claro que es un mero dispositivo notacional, ya que todas nuestras funciones, salvo aviso explícito en contrario, tienen exactamente un argumento.

Las funciones sobreyectivas π^L y π^R, o funciones de proyección, que conjuntamente actúan como una inversa de $\langle x.y\rangle$, permiten descodificar el contenido de un número, seleccionando individualmente cada una de las componentes del par: $\langle \pi^L(x).\pi^R(x)\rangle = x$. Existen funciones similares para tamaños mayores de tupla. Convenimos que $\pi^L(0) = 0$ y $\pi^R(0) = 0$.

Resulta preciso conocer el tamaño de la tupla codificada; es decir, por ejemplo, cuando $\langle x.y.z\rangle = \langle u.v\rangle$, con $v \neq 0$, la segunda componente, y o v, habrá de calcularse de manera distinta en cada caso. De hecho, con nuestra definición, en este ejemplo $v = \langle y.z\rangle$. Así pues, para cada tamaño de tupla $m > 1$, hay m funciones de proyección distintas.

Para operar con secuencias de naturales de longitud finita pero arbitraria, usamos una notación similar, pero separando ahora sus componentes por comas. Definimos $\langle x\rangle = \langle x.0\rangle$, $\langle x, y\rangle = \langle x.\langle y\rangle\rangle = \langle x.\langle y.0\rangle\rangle$ y, en general, para $m > 1$, $\langle x_0, \ldots, x_{m-1}\rangle = \langle x_0.\langle x_1, \ldots, x_{m-1}\rangle\rangle$. De manera natural, el 0 codifica, según este esquema, la secuencia vacía $\langle\rangle$.

Reservamos el símbolo π^* para la función de tupla sufijo: interpreta su argumento como un par $\langle y.k\rangle$ e y como una tupla de al menos k componentes, y devuelve la tupla que consta de las componentes de y de la k-ésima en adelante: para $0 \leq k \leq m$, $\pi^*(\langle\langle x_0, \ldots, x_{m-1}\rangle.k\rangle) = \langle x_k, \ldots, x_{m-1}\rangle$, que es $\langle\rangle$ si $k = m$. Se define formalmente por: $\pi^*(0) = 0$, $\pi^*(\langle x.0\rangle) = x$, y, para $k > 0$, $\pi^*(\langle x.k\rangle) = \pi^*(\langle \pi^R(x).k - 1\rangle)$.

Reservamos el símbolo π para la función general de proyección: una función parcial que interpreta su argumento también como un par $\langle y.k\rangle$ e y como una tupla de al menos k componentes, y devuelve la componente k-ésima de y: para $0 \leq k < m$, $\pi(\langle\langle x_0, \ldots, x_{m-1}\rangle.k\rangle) = x_k$. Se define formalmente por: $\pi(0) = 0$, $\pi(\langle x.k\rangle) = \pi^L(\pi^*(\langle x.k\rangle))$.

Más adelante necesitaremos las propiedades siguientes, que el lector con inclinaciones matemáticas no tendrá dificultad en demostrar por inducción:

Propiedad de monotonía por componentes: vista como función de uno cualquiera de sus argumentos, y manteniendo los demás fijos,

$$x_i < y \Rightarrow \langle x_0, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_{m-1}\rangle < \langle x_0, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_{m-1}\rangle.$$

Propiedad de codificación incremental: cualquiera que sea $m \geq 1$,

$$\forall x_0, \ldots, x_{m-1}, (x_i < \langle x_0, \ldots, x_i, \ldots, x_{m-1}\rangle).$$

3 Las funciones recursivas parciales

You're the snake charmer, baby.
And you're also the snake.
You're a closed circuit, baby.
You've got the answers in the palms of your hands.

Laurie Anderson: Closed Circuit

Denominaremos "funciones básicas" a las siguientes: la constante 1, la identidad, la función de tupla sufijo π^*, la función general de proyección π, la suma $\langle x.y\rangle \to x + y$, el producto $\langle x.y\rangle \to x * y$ y la diferencia modificada $\langle x.y\rangle \to x \dot{-} y$.

A partir de estas funciones básicas, y combinándolas mediante composición y formación de pares, es posible obtener las siguientes funciones:

1. cada función constante;
2. para cada natural i, la función que asocia a x el valor $\langle x.i\rangle$;
3. las funciones anterior `ant`(x) y signo `sg`(x) definidas en la primera página;
4. las dos proyecciones π^L y π^R;
5. las operaciones de conjunción, disyunción y negación operan con los naturales 0 y 1 como si fuesen valores booleanos: $\langle x.y\rangle \to x \wedge y$, $\langle x.y\rangle \to x \vee y$ y $x \to \neg x$;
6. los predicados de comparación según el orden y de igualdad;
7. la función que a x asocia $\langle x.x\rangle$;
8. la función que a $\langle x.y\rangle$ asocia $\langle y.x\rangle$.

Sea A un conjunto de naturales, y sean f y g funciones totales. Combinándolas con las básicas, y con χ_A, mediante composición y formación de pares, podemos obtener h:

$$h(x) = \begin{cases} f(x) & \text{si } x \in A, \\ g(x) & \text{si } x \notin A. \end{cases}$$

Una manera de lograrlo es ir aplicando formación de pares y composición hasta obtener la expresión:

$$\chi_A(x) * f(x) + (1 - \chi_A(x)) * g(x).$$

Por supuesto, es posible generalizar esta construcción a más de dos casos. Es instructivo considerar qué ocurre cuando se intenta aplicar la misma solución cuando el rol de f y/o g lo juegan funciones parciales: la definición por casos resultará ser válida porque se puede argumentar de otra manera alternativa, pero esta argumentación ya no lo es.

3.1 La operación de minimización

El operador de minimización sobre una función total g es similar a un programa de búsqueda lineal. Decimos que la función parcial ψ se define por minimización sobre la función total g, y lo denotamos $\psi(x) = \mu z[g(x,z) = 1]$, si

$$\psi(x) = \begin{cases} \min\{z \mid g(\langle x.z\rangle) = 1\} & \text{si este conjunto es no vacío,} \\ \uparrow & \text{en otro caso.} \end{cases}$$

Definimos las funciones recursivas parciales como aquellas que se pueden obtener a partir de las funciones básicas, combinándolas mediante composición, formación de pares y minimización. Ejemplos de funciones que podemos demostrar que son recursivas parciales usando el operador de minimización son:

1. la función completamente indefinida y
2. las funciones de división entera por defecto y de resto de dicha división.

Además: la inversa f^{-1} de una función recursiva total inyectiva f es recursiva parcial.

En caso de que f no sea inyectiva, pueden existir varias funciones distintas que, en un sentido determinado, "actúan como una inversa"; es posible garantizar la recursividad parcial de una de ellas. La misma propiedad se podrá extender a funciones parciales.

Recordemos que las funciones características, que son funciones totales que sólo toman los valores 0 y 1, se pueden ver como predicados, o funciones booleanas. Sea P un predicado recursivo. El esquema de minimización nos permite argumentar que también lo son:

$$P'(\langle x.t\rangle) = 1 \iff \exists y(y \leq t \wedge P(\langle x.y\rangle) = 1),$$

$$P''(\langle x.t\rangle) = 1 \iff \forall y(y \leq t \Rightarrow P(\langle x.y\rangle) = 1).$$

Decimos que estos predicados se obtienen por cuantificación existencial acotada y por cuantificación universal acotada, respectivamente.

Cuando consideramos familias de funciones parciales, podemos permitir además la siguiente variante de cuantificación existencial: para todo x, $P'''(x)\uparrow$ o bien $P'''(x) = 1$ y, también para todo x,

$$P'''(x) = 1 \iff \exists y(P(\langle x, y\rangle) = 1).$$

donde quizá conviene insistir en que P es total. De manera análoga, se puede definir el predicado que se obtiene mediante cuantificación universal no acotada. Aún no tenemos suficiente artillería para argumentarlo, pero lo cierto es que, a partir de un predicado recursivo total P, el que definimos sobre él por cuantificación universal no acotada puede no ser recursivo. Por ese motivo, no entramos en detalles sobre esta construcción.

La definición de minimización puede extenderse al caso en que partimos de una función parcial ψ. Si imaginamos que intentamos calcular su valor por una búsqueda lineal, podemos intuir que no conseguiremos respuesta en dos ocasiones: cuando el objeto buscado no existe, o cuando uno inferior a él provoca una búsqueda indefinida que no para, y así nos hace dudar de cuál es realmente el mínimo.

En concreto, decimos que la función parcial ψ se define por minimización sobre la función parcial ϕ si

$$\psi(x) = \begin{cases} \min\{z \mid \phi(\langle x.z\rangle) = 1 \wedge \forall w < z\,(\phi(\langle x.w\rangle)\downarrow)\} & \text{si este conjunto es no vacío,} \\ \uparrow & \text{en otro caso.} \end{cases}$$

Es posible argumentar que este operador no trae aparejadas nuevas funciones recursivas parciales: si nos restringimos a usar la minimización únicamente sobre funciones totales, no por ello perdemos funciones.

3.2 Recursión primitiva

En la definición original de las funciones recursivas parciales aparecía un operador de recursividad, como aún refleja el nombre. En la definición alternativa que hemos dado, éste no aparece. Argumentamos seguidamente cómo obtener una de sus variantes; nos facilitará la programación de otras importantes funciones más adelante.

Decimos que s se define a partir de las funciones totales f y g por recursión primitiva sobre la secuencia de valores (nombre que abreviaremos a "por recursión primitiva", o incluso a "por recursión"; en lengua inglesa "course-of-values primitive recursion") si hay un k tal que se cumplen las siguientes ecuaciones:

$s(x) = g(x)$ para $x \leq k$

$s(x) = f(\langle x, s(x-1), \ldots, s(1), s(0)\rangle)$ para $x > k$

Propiedad de recursión primitiva: Si f y g son recursivas totales y s se define a partir de ellas por recursión primitiva sobre la secuencia de valores, entonces s también es recursiva total.

(Es posible permitir funciones parciales en lugar de f y g pero notación y argumentación se complican innecesariamente.)

Bosquejamos una argumentación de por qué se puede contar con esta propiedad; seguiríamos los pasos siguientes:

1. Sean f y g funciones recursivas totales, y k un natural. El siguiente predicado, que denotaremos $R_{f,g,k}(\langle x.y\rangle)$, es recursivo, donde para facilitar la lectura denotamos $y = \langle y_x, \ldots, y_0\rangle$:

$$\forall t \leq x(t \leq k \Rightarrow y_t = g(t) \wedge t > k \Rightarrow y_t = f(\langle t, y_{t-1}, \ldots, y_0\rangle))$$

2. A partir del apartado anterior, definimos $\phi(x) = \mu y[R_{f,g,k}(\langle x.y\rangle) = 1]$, donde f y g son de nuevo funciones recursivas totales, y k un natural. Razonando por inducción, podemos ver que $\phi(x)$ codifica una tupla de longitud $x+1$, de la forma $\langle s(x), s(x-1), \ldots, s(1), s(0)\rangle$.

3. Y, finalmente, consideramos $\pi(\langle \phi(x).0\rangle)$, que resulta ser $s(x)$.

Como ejemplo de aplicación de la recursión primitiva, podemos argumentar que la función factorial es recursiva.

La recursión primitiva es importante por varios motivos. Algunos aparecerán más adelante. Uno en el que no entraremos más allá de esta breve alusión consiste en que proporciona una

definición alternativa de las funciones recursivas parciales, de hecho, la original. Si en lugar de trabajar con tuplas combinadas en un único número natural "estratificamos" las funciones recursivas parciales en función de su número de argumentos y generalizamos la operación de composición, encontramos que nos basta la constante cero, la función sucesor y las proyecciones, como funciones básicas, para obtener todas las funciones recursivas parciales mediante composición, recursión primitiva y minimización.

4 La función universal

> Without being able to say how, anymore than a computer can explain its own processes, she saw them as they were and not as they tried to seem.
>
> *John Fowles: The French Lieutenant's Woman*

Los dispositivos que tenemos ahora para codificar tuplas en un único número nos van a permitir el paso clave en la construcción de "intérpretes" o funciones universales: representar cada programa de una manera usable por otro programa.

En las máquinas de Turing, hemos visto diversas maneras de representar la matriz de transición con un código basado en caracteres que pueden aparecer en la cinta de otra máquina de Turing. Del mismo modo, un programa en Python no es sino un "string" que puede ser leído y procesado por otro programa en Python (y podemos sustituir aquí Python por cualquier otro lenguaje de programación).

Las funciones recursivas parciales trabajan sobre números naturales; así pues, para que una de estas funciones "procese otra", lo que conviene es asignarles un número a cada una. Ahora explicamos cómo hacerlo de manera que el número de cada función explique la propia construccion de la función. Dado que el primero en emplear este tipo de idea para asignar números a objetos sintácticamente complicados (en su caso concreto, fórmulas de la Lógica de primer orden) fue Gödel, este proceso recibe frecuentemente un nombre alusivo a él.

4.1 Gödelización

Vamos, pues, a asignar a cada una de las funciones recursivas parciales un código numérico, de tal manera que ese número nos justifique que, en efecto, la función correspondiente es recursiva parcial. Denotaremos por ϕ_i la función recursiva parcial a la que corresponda el número i.

En primer lugar, consideramos las siete funciones básicas. Asociaremos a la j-ésima de estas funciones el código $\langle 0.j\rangle$. En particular, concretamos los códigos $\langle 0.0\rangle = 1$ para la constante 1, $\langle 0.1\rangle = 2$ para la identidad, $\langle 0.2\rangle = 4$ para la función de tupla sufijo π^*, $\langle 0.3\rangle = 7$ para la función general de proyección π, $\langle 0.4\rangle = 11$ para la suma, $\langle 0.5\rangle = 16$ para el producto y $\langle 0.6\rangle = 22$ para la diferencia modificada. Si deseáramos añadir más funciones a la familia de las básicas, no habría problema en llegar a números más altos.

En segundo lugar, acordamos que el código $i = \langle 1.\langle j.k\rangle\rangle$ corresponde a la función $\phi_i = \phi_j \circ \phi_k$. Análogamente, el código $i = \langle 2.\langle j.k\rangle\rangle$ corresponde a la función $\phi_i = \langle \phi_j.\phi_k\rangle$ que se define por formación de pares, y el código $i = \langle 3.j\rangle$ corresponde a la función ϕ_i que se define por minimización sobre ϕ_j.

Finalmente, acordamos que si i no corresponde a ninguno de los casos descritos, entonces ϕ_i es la función completamente indefinida. En particular, teniendo en cuenta el esquema que usamos para formar pares, ϕ_0 es la función completamente indefinida: $\forall x(\phi_0(x)\uparrow)$.

Repetimos una vez más, ahora que la numeración de las funciones recursivas parciales es más concreta, la intuición explicada antes y que es importante comprender: el código numérico que asignamos a una función explica cómo calcularla. En ese sentido, el número es, a la vez, el programa o algoritmo que justifica que la función es, en efecto, intuitivamente calculable.

Es importante observar que, en principio, cada función puede tener varios códigos; de hecho, tendrá siempre un número infinito de ellos, ya que, por ejemplo, para cada valor de k, la identidad $\phi(x) = \phi \circ \pi^R(\langle k.x\rangle)$ da lugar a un número de código distinto para la misma función $\phi(x)$.

Hay que distinguir, por tanto, la propia función ϕ_i del programa que la calcula, codificado por i. Dado i, habrá otro programa, codificado por un número $j \neq i$, tal que $\phi_i = \phi_j$. Por ello, al hablar de las propiedades de ϕ_i, a veces se tratará de la función propiamente dicha, y otras se tratará del programa codificado por i. El lector ha de ser capaz de discernir cuándo se trata de una propiedad del programa y cuándo de una propiedad de la función.

4.2 El predicado de Kleene y la universalidad

El predicado T de Kleene es la piedra angular sobre la que construiremos la función universal y sus variantes. Formaliza la propiedad que podemos describir, de manera intuitiva, como sigue: "la función de índice i, sobre el argumento x, evalúa a y, y ello se puede justificar usando una secuencia de valores intermedios, globalmente acotada por t". A este nivel intuitivo, podemos asimilar el valor t a una cota sobre el uso de recursos disponibles para calcular $\phi_i(x)$: si no nos basta con, imaginemos, tiempo t para completar el cálculo, el predicado se hace 0. De hecho, se pueden obtener predicados T de Kleene, siempre con parecidas propiedades, a partir de muchos otros modelos de cálculo, y en algunos de ellos el rol del parámetro t es exactamente "tiempo" o número de instrucciones elementales realizadas.

Para facilitar la lectura, escribimos $f(x, y)$ por $f(\langle x.y\rangle)$ o $g(x, y, z, t)$ por $g(\langle x.y.z.t\rangle) = g(\langle x.\langle y.\langle z.t\rangle\rangle\rangle)$. El predicado $T(i, x, y, t)$ se define como sigue: $T(i, x, y, t) = 1$ se da

- si ϕ_i es básica, digamos, $i = \langle 0.j\rangle$, cuando $t \geq x$ y $\phi_i(x) = y$;
- si $\phi_i = \phi_j \circ \phi_k$, cuando

$$\exists s \leq t(s = \langle z.y.p.q\rangle \wedge T(k, x, z, p) \wedge T(j, z, y, q));$$

- si $\phi_i = \langle \phi_j.\phi_k\rangle$, cuando

$$y = \langle z.w\rangle \wedge \exists s \leq t(s = \langle p.q\rangle \wedge T(j, x, z, p) \wedge T(k, x, w, q));$$

- si $\phi_i = \mu\phi_j$, cuando

$$\exists s \leq t(\langle\langle x.y\rangle.s\rangle \leq t \wedge s = \langle\langle z_0.s_0\rangle, \ldots, \langle z_y.s_y\rangle\rangle \wedge$$

$$\forall m \leq y\, T(j, \langle x.m\rangle, z_m, s_m) \wedge \forall m < y\, (z_m \neq 1) \wedge z_y = 1).$$

Con arreglo a esta definición, nos fijamos en las siguientes observaciones:

- En cada uno de los casos, se está definiendo $T(u)$ en términos de T sobre valores inferiores a u, debido a las propiedades de codificación incremental y de monotonía por componentes.
- Todos los valores v tales que $T(v)$ es relevante en el cálculo de $T(u)$ se pueden calcular por funciones recursivas totales a partir de u.
- Entonces, el que T sea un predicado recursivo total se deduce del esquema de recursión primitiva.

Además, se cumple la propiedad siguiente: para todo t, t' con $t < t'$, si $T(i, x, y, t) = 1$, entonces $T(i, x, y, t') = 1$. Intuitivamente, esto se puede leer: "con más tiempo, no perdemos los cálculos que ya nos daba tiempo a hacer", es decir, para todo z, con un t suficientemente alto, $T(i, x, z, t)$ si y sólo si $z = \phi_i(x)$. Más precisamente:

- Para todo t, si $T(i, x, y, t) = 1$ entonces $\phi_i(x)\downarrow$ y además $\phi_i(x) = y$
- Para todo i y x, $\phi_i(x)\downarrow$ si y sólo si $\exists t \exists y (T(i, x, y, t) = 1)$

Con lo que podemos obtener:

Propiedad de la función universal La función recursiva parcial definida por minimización y composición en términos de π^R y T, como

$$\phi_{\mathbf{u}}(i, x) = \pi^R(\mu\langle t.y\rangle[T(i, x, y, t) = 1])$$

cumple la siguiente igualdad: para todo i y x, $\phi_{\mathbf{u}}(i, x) = \phi_i(x)$.

4.3 Indecidibilidad

Ahora podemos considerar el dominio de la función universal, así como el de una variante suya. Ambos reciben el nombre de "problema de la parada" (de hecho, es posible demostrar que son equivalentes en un sentido muy estricto).

Concretamente, el "problema de parada" es Dom $\phi_{\mathbf{u}}$, el dominio de la función universal; es decir, el conjunto $K_0 = \{\langle i.x\rangle \mid \phi_u(\langle i.x\rangle)\downarrow\} = \{\langle i.x\rangle \mid \phi_i(x)\downarrow\}$. El par $\langle i.x\rangle$ pertenece a K_0 si y sólo si la función de índice i está definida sobre x. La expresión "parada" proviene del concepto análogo en máquinas de Turing: ¿parará la máquina de código i cuando calcule sobre la entrada x?

Por extensión, llamamos también problema de parada al conjunto $K = \{i \mid \phi_i(i)\downarrow\}$. Claramente $i \in K \iff \langle i.i\rangle \in K_0$.

Supongamos que K es decidible, es decir, χ_K es recursiva total; veremos que esta suposición nos lleva a contradicción. A partir de esta función construimos otra, que cumple: $\phi(x) = 0$ si $x \notin K$, y $\phi(x)\uparrow$ si $x \in K$. La podemos plantear así:

$$\phi(x) = \mu z[\pi^L(\langle \chi_K(x).z\rangle) = 0]$$

Si todas las funciones involucradas son recursivas totales, ϕ es recursiva parcial y, por tanto, le corresponde un índice: existe un j tal que, para todo x, $\phi_j(x) = 0$ si $x \notin K$, y $\phi_j(x)\uparrow$ si $x \in K$.

Podemos considerar el caso $x = j$. Estas últimas igualdades nos dicen, entonces, que $\phi_j(j)\downarrow$ si y sólo si $j \notin K$. Pero la definición de K nos dice, justamente, lo contrario: $j \in K$ si y sólo si $\phi_j(j)\downarrow$. La única explicación de esta contradicción es que la función χ_K no es recursiva, es decir:

Teorema de Indecidibilidad del Problema de Parada. K es indecidible.

El mismo resultado se da para la otra variante del problema de parada, por supuesto, ya que $i \in K \iff \langle i.i\rangle \in K_0$, por lo que si K_0 fuese decidible, K también lo sería.

5 Reducibilidad

> I'm guided by a signal in the heavens.
> I'm guided by this birthmark on my skin.
> I'm guided by the beauty of our weapons.
> First we take Manhattan, then we take Berlin.
>
> *Leonard Cohen: First We Take Manhattan*

El planteamiento del último párrafo de la sección anterior se aplica para argumentar la indecidibilidad de muchos otros problemas. Existen diversas variantes de reducibilidad, con propiedades diferentes, pero aquí nos centramos únicamente en la llamada m-reducibilidad: L es reducible a L' si existe una función recursiva total f tal que, para todo x, $x \in L \iff f(x) \in L'$; o, lo que es lo mismo, $L = f^{-1}(L')$ o, también, $\chi_L = \chi'_L \circ f$. Lo denotamos $L \leq_m L'$.

Así, si $L \leq_m L'$ y L' es decidible, es decir, si su función característica es recursiva total, deducimos mediante el cierre por composición que también lo es L, porque $\chi_L = \chi'_L \circ f$.

Razonando a la inversa, si $L \leq_m L'$ y L es indecidible, también lo es L'. Ahora tenemos un problema indecidible, a saber, el problema de parada K. Siempre que encontremos un problema L' y una función f recursiva total que cumpla $x \in K \iff f(x) \in L'$ para todo x, deducimos que L' también es indecidible.

5.1 Parametrización y evaluación parcial

La herramienta principal para obtener este tipo de funciones es la parametrización, también llamada evaluación parcial. A partir de una función que interpreta su argumento como un par, nos permite fijar la primera componente del par y obtener una nueva función que depende de la segunda. (La función `partial` del módulo `functools` de Python tiene exactamente este comportamiento.)

Teorema de Parametrización (versión simplificada o "s-1-1"). Existe una función s_1 recursiva total tal que para todo i, x, y, $\phi_i(x, y) = \phi_{s_1(i,x)}(y)$.

(Recordemos, una vez más, que todas nuestras funciones tienen un solo argumento y que esta última expresión es una simplificación de $\phi_i(\langle x.y\rangle) = \phi_{s_1(\langle i.x\rangle)}(y)$.)

En concreto, esta función se construye del modo siguiente. Sabemos que el índice de la composición de dos funciones recursivas tiene asociado un número de Gödel específico; en nuestro convenio, $\langle 1.\langle p.q\rangle\rangle$ para la compuesta de las funciones de números p y q. Lo abreviamos $c(\langle p.q\rangle)$, claramente recursiva total. También tenemos que las funciones $f(y) = \langle 0.y\rangle$ y $g(\langle x.y\rangle) = \langle x{+}1.y\rangle$ son recursivas totales, como ya sabemos: $\phi_p = f$ y $\phi_q = g$ para los correspondientes números

de Gödel p y q. Mediante recursión primitiva, tenemos también h definida como: $h(0) = p$ y $h(n+1) = c(\langle q.h(n)\rangle)$. Entonces, hacemos $s_1(\langle i.x\rangle) = c(\langle i.h(x)\rangle)$.

Se comprueba sin dificultad, ahora, que $\phi_{h(x)}(y) = \langle x.y\rangle$ y que, por tanto, $\phi_{s_1(\langle i.x\rangle)}(y) = \phi_i(\langle x.y\rangle)$, tal como hemos enunciado.

5.2 Conjuntos índex

Consideremos ahora el conjunto de los índices de funciones totales, $\{i \mid \forall x\, \phi_i(x)\downarrow\}$. Será nuestro siguiente ejemplo de problema indecidible.

Partimos de la siguiente función recursiva parcial: $\rho(\langle x.y\rangle) = \phi_{\mathbf{u}}(\langle x.x\rangle)$, y sea j tal que $\phi_j = \rho$. Para ese j, se puede comprobar ahora fácilmente que $x \in K$ si y sólo si $\phi_{s_1(\langle j.x\rangle)}$ es total; es decir, definiendo $f(x) = s_1(\langle j.x\rangle)$ para ese j, $x \in K \iff f(x) \in \{i \mid \forall x\, \phi_i(x)\downarrow\}$. Por tanto, $\{i \mid \forall x\, \phi_i(x)\downarrow\}$ es indecidible.

Es decir: entre las funciones recursivas parciales ϕ_i, algunas son totales, otras no; y no es posible decidir cual de los dos casos es para un índice i mediante un algoritmo.

El fenómeno que subyace es más general. Consideremos cualquier pregunta que se pueda hacer sobre funciones recursivas parciales que sólo se refiera a la función, y no al índice particular que se considere.

Teorema de Rice. Sea $\mathcal{F}$ un conjunto arbitrario de funciones recursivas parciales. Consideramos el conjunto A de los índices de programas que calculan funciones de $\mathcal{F}$:

$$A = \{x | \phi_x \in \mathcal{F}\}$$

y supongamos que la propiedad que defina $\mathcal{F}$ no es trivial, es decir, hay funciones recursivas parciales que están en $\mathcal{F}$ (por tanto $A \neq \emptyset$) y otras que no lo están (por tanto $A \neq \mathbb{N}$). Entonces, el conjunto A es indecidible.

Los conjuntos A definidos de esta manera se llaman *conjuntos índex*. Motivemos y expliquemos este enunciado. Al estudiar programas concretos, se pueden plantear muchas propiedades interesantes a analizar. Algunas de ellas dependen del texto específico del programa como, por ejemplo, el número de bucles que contiene. Otras se refieren en esencia a la función que calcula el programa, o sea, a la relación entre su salida y su entrada; por ejemplo, el ser o no un programa correcto para calcular el factorial de un número dado. La característica de este segundo tipo es que no le afecta el cambio del programa por otro distinto pero que se comporte de manera exactamente igual.

Una propiedad sobre programas siempre da lugar a un conjunto de números naturales: aquellos que codifican un programa que cumple la propiedad en cuestión. Por ejemplo, el conjunto K corresponde a los programas que tienen la propiedad de parar cuando el dato es su propio número. Ésta es una propiedad del programa, no sólo de la función que éste calcula: es posible tener dos números distintos, i y j, tales que $\phi_i = \phi_j$, $\phi_i(i)\downarrow$, pero $\phi_i(j) = \phi_j(j)\uparrow$. A pesar de representar la misma función, $i \in K$ pero $j \notin K$.

Consideremos ahora otra propiedad diferente: la de no parar nunca, sobre ningún dato. Corresponde al conjunto V formado por los números de programa de dominio vacío, es decir, índices de la función completamente indefinida. Si el programa de índice i no para nunca, ϕ_i es la función completamente indefinida. Por tanto, si $\phi_j = \phi_k$, o bien ninguno de los dos para nunca, o bien ambos paran con algún dato; luego: o la propiedad se cumple para los dos programas j y k,

o falla para ambos. No puede ser que uno esté en V y el otro no: V es un conjunto índex. Un ejemplo adicional lo tenemos en el conjunto de los índices de funciones totales, que también lo es.

El teorema de Rice nos indica, por tanto, que, si consideramos propiedades de los programas que sólo dependan de la función que calculan (relación datos/resultados), las únicas decidibles son triviales. En particular, V es indecidible.

Como ilustración adicional, recordemos el funcionamiento de `jutge.org`. Sabemos que juzga nuestros programas comparándolos con un programa fijo en una cierta cantidad de juegos de pruebas. ¿No podría implementar algo más sofisticado, comprobando, de alguna manera, que el programa que se somete hace exactamente lo mismo que la solución patrón? El teorema de Rice nos da la respuesta: no es posible.

Argumentemos el teorema de Rice. Recordemos que ϕ_0 es la función completamente indefinida. Si es necesario, permutemos los nombres de A y de su complementario, de forma que $0 \notin A$. Como $A \neq \emptyset$, podemos elegir $a \in A$. Partimos de la siguiente función recursiva parcial: $\rho(\langle x.y\rangle) = \phi_a(y) * \mathtt{sg}(1 + \phi_{\mathbf{u}}(\langle x.x\rangle))$, y sea de nuevo j tal que $\phi_j = \rho$.

Si $x \in K$, entonces $\phi_{\mathbf{u}}(\langle x.x\rangle)\downarrow$, y $\mathtt{sg}(1+\phi_{\mathbf{u}}(\langle x.x\rangle)) = 1$, con lo que $\phi_{s_1(\langle j.x\rangle)}(y) = \phi_j(\langle x.y\rangle) = \phi_a(y)$ para todo y; por la propiedad de A, al ser la misma función, $s_1(\langle j.x\rangle) \in A$.

Si $x \notin K$, entonces $\phi_{\mathbf{u}}(\langle x.x\rangle)\uparrow$ y, con él, $\phi_{s_1(\langle j.x\rangle)}(y)\uparrow$ cualquiera que sea y. Así pues $\phi_{s_1(\langle j.x\rangle)}$ es la función completamente indefinida ϕ_0 y, por las propiedades de A, $s_1(\langle j.x\rangle) \notin A$.

Así pues, definiendo $f(x) = s_1(\langle j.x\rangle)$ tenemos que $x \in K \iff f(x) \in A$ y, por tanto, A es indecidible.

Como ejemplo de aplicación del teorema de Rice, tenemos que el conjunto $\{i \mid \phi_i \text{ es constante }\}$ es indecidible. Función constante significa que hay un valor $k \in \mathbb{N}$ tal que $\forall x\, \phi_i(x) = k$.

5.3 Los teoremas de recursión

Otra de las aplicaciones de parametrización consiste en una construcción que permite a una función recursiva parcial conocer su propio número. En un sentido más general, es como decir que un programa puede conocer su propio texto. Por supuesto, una manera de usar su propio texto que queda disponible para un tal programa es llamarlo, es decir, "llamarse a sí mismo": nos abre la puerta a los programas recursivos que conocemos desde primer curso.

Alternativamente, un programa que conoce su propio texto puede, simplemente, escribirlo como salida. Se trataría de un programa que "se escribe a sí mismo". (Si deseas intentarlo, no lo busques en la Wikipedia, que lo encontrarás antes de tiempo.)

Veamos cómo se argumenta la existencia de este tipo de programas. De hecho, existen varios "teoremas de recursión" con características variadas; aquí indicamos solamente la variante de Rogers sobre la versión de Kleene:

Teorema de recursión. Sea f una función recursiva total. Existe un número natural n tal que $\phi_{f(n)} = \phi_n$.

Se aplica frecuentemente a una f obtenida por parametrización. Como ejemplo, podemos argumentar que existe un programa que calcula una función constante cuyo valor es el propio número del programa, como acabamos de anunciar. Si j es el número de Gödel de la proyección izquierda, $\phi_j(\langle x.y\rangle) = x$, tomamos $f(x) = s_1(\langle j.x\rangle)$: por el teorema de recursión, hay un n tal que, para todo y,

$$\phi_n(y) = \phi_{f(n)}(y) = \phi_{s_1(\langle j.n\rangle}(y) = \phi_j(\langle n.y\rangle) = n.$$

De manera similar, podemos obtener el número de una función recursiva parcial m tal que únicamente esté definida en su propio número: el dominio de ϕ_m se reduce al conjunto $\{m\}$; o bien el número de una función recursiva total m que "se identifica a sí misma": $\phi_m(m) = 1$ y $\phi_m(y) = 0$ si $y \neq m$.

La argumentación del teorema de recursión está relacionada con el combinador Y del lambda-cálculo, que quizá podamos explicar más adelante. Su esencia es como sigue: empezamos aplicando parametrización para obtener una función s tal que:

$$\phi_{s(x)}(y) = \phi_{\phi_x(x)}(y),$$

que sólo está definida si $\phi_x(x)\downarrow$ (o sea, $x \in K$) y, además, el valor $z = \phi_x(x)$ es tal que $\phi_z(y)\downarrow$.

La función dada f, de la que buscamos el punto fijo, es recursiva total y, por tanto, su composición con s también lo es. Hay un índice m tal que $\phi_m = f \circ s$. En ese punto, se comprueba que el número $n = s(m)$ cumple las condiciones deseadas.

6 Enumerabilidad recursiva

> Me dejaste masticando la aritmética fatal
> de ir sumando los segundos esperando la señal.
>
> *Stupendams: La señal*

A partir del primer problema indecidible, K, hemos ido identificando otros. Aparecen de inmediato preguntas adicionales relacionadas: ¿Son todos los problemas indecidibles "igual de indecidibles"? K, en tanto que indecidible, ¿"cuánto de lejos está" de ser decidible? No entraremos en la fascinante teoría que se desarrolla por esta vía; pero sí explicaremos un poco más sobre las propiedades de K, debido a que la teoría de la NP-completitud, que trataremos a continuación, nace de analogías con estas propiedades.

En la sección 3.1 hemos mencionado predicados recursivos parciales: indican pertenencia a un conjunto (su dominio) con un 1, pero no indican la no pertenencia, ya que en el resto de los casos quedan indefinidos debido a un cuantificador existencial no acotado que lleva a una búsqueda lineal que nunca termina.

Por ejemplo: $i \in K \iff \exists \langle y.t \rangle\, T(i, i, y, t)$, por las propiedades del predicado de Kleene T. Así, podemos obtener K como dominio de un predicado recursivo parcial, es decir, por cuantificación existencial de un predicado decidible.

Los dominios de predicados recursivos parciales resultan poderse definir de muchas maneras. Las siguientes propiedades son equivalentes:

1. A es el dominio de un predicado recursivo parcial.
2. A es el dominio de alguna función recursiva parcial (predicado o no).
3. A es la imagen de alguna función recursiva parcial.
4. A es la imagen de alguna función recursiva total (o bien es vacío).
5. A se obtiene por cuantificación existencial sobre un predicado decidible.

Cuando un conjunto cumple estas propiedades, decimos que es *enumerable recursivamente*. El nombre alude al punto 4: una función recursiva f va enumerando todos y cada uno de los elementos del conjunto, $f(0)$, $f(1)$, $f(2)$, ...

6.1 Completitud

De nuevo, la m-reducibilidad tiene un rol en el estudio de los conjuntos enumerables recursivamente. Es posible demostrar que:

Si $L \leq_m L'$ y L' es enumerable recursivamente, entonces L también lo es; es decir, si $L \leq_m L'$ y L no es enumerable recursivamente, entonces L' tampoco lo es.

Intuitivamente, si $L \leq_m L'$, indica que L' es "al menos tan difícil como L", porque una solución de L' proporciona una solución para L tanto en términos de decidibilidad como en términos de enumerabilidad recursiva. En ese sentido, K es "el más difícil" de los enumerables recursivamente, porque:

Teorema . K es completo en los enumerables recursivamente para la m-reducibilidad, es decir, todo conjunto A enumerable recursivamente se reduce a K: $A \leq_m K$

En efecto, si $A = \text{Dom } \phi_j$, por parametrización, podemos obtener una funcion f tal que:

$$\phi_{f(i)}(x) = \phi_j(i)$$

Entonces, si $i \in A = \text{Dom } \phi_j$, obtenemos que $\phi_{f(i)}(x)\downarrow = \phi_j(i)$ y es, por tanto una función constante total ya que el valor que da es independiente de x. En particular, $\phi_{f(i)}(f(i))\downarrow$ y, por tanto, $f(i) \in K$. Viceversa, si $i \notin A$, $\phi_j(i)\uparrow$, $\phi_{f(i)}$ es la función completamente indefinida, $\phi_{f(i)}(f(i))\uparrow$ y, por tanto, $f(i) \notin K$. Luego $i \in A \iff f(i) \in K$ y por tanto $A \leq_m K$.

Todas estas nociones tendrán un correlato en la teoría de la NP-completitud, que nos proporciona valiosa información sobre cuándo podemos, y cuándo no, evitar programas exponencialmente lentos para resolver problemas combinatorios.

6.2 Complementación: casos que no son enumerables recursivamente

A partir de un predicado recursivo total, podemos cambiarlo para que entre en un bucle infinito en caso de dar 1, por lo cual los conjuntos decidibles son enumerables recursivamente. Como el complementario de un conjunto decidible es decidible, también es enumerable recursivamente. La observación importante en este punto es:

Teorema de complementación. Un conjunto es decidible si y solo si tanto él como su complementario son enumerables recursivamente.

La idea de la demostración es como sigue: si $x \in A$ cuando existe un y "que lo testimonia" (mediante un predicado decidible) y $x \in \overline{A}$ cuando existe un z "que lo testimonia" (mediante otro predicado decidible), aplicamos minimización y disyunción para buscar a la vez el y y el z. Sabemos que o bien $x \in A$ o bien $x \in \overline{A}$, así pues, o bien encontraremos el y o bien el z, la minimización siempre estará definida y nos dirá de cuál de los dos casos se trata.

Obtenemos, como consecuencia, el primer ejemplo de conjunto no enumerable recursivamente: el complementario de K, denotado $\overline{K}$, ya que, si lo fuera, como K también lo es, ambos serían decidibles, y ya sabemos que no es así. Existen extensiones del teorema de Rice (en las que no entraremos) que permiten argumentar que muchos conjuntos índex no sólo no son decidibles, sino que ni siquiera son enumerables recursivamente.

6.3 El "Entscheidungsproblem"

Consideramos la lógica de primer orden apropiada para hablar de la aritmética de los números naturales: símbolos de función para el sucesor, la suma y la multiplicación; predicados para igualdad y desigualdad; y, por supuesto, la constante cero. Cada fórmula puede ser cierta o no en el modelo estándar: los números naturales.

Cada número natural n se puede representar mediante el término correspondiente a n aplicaciones de la función sucesor sobre la constante cero. Denotamos $\mathtt{n}$ ese término.

Un conjunto de naturales A es definible en la aritmética si existe una fórmula que lo define, en el sentido siguiente: existe una fórmula $\alpha = \alpha(x)$ con una variable libre x tal que la fórmula $\alpha(\mathtt{n})$, que se obtiene al sustituir todas las menciones a x en α por el término $\mathtt{n}$, es cierta en $\mathbb{N}$ si y sólo si $n \in A$.

Una de las contribuciones principales de Gödel fue demostrar que los dominios de las funciones recursivas parciales son definibles. Por supuesto, en su día, este hecho se expresaba de otra manera (mediante cuantificación y funciones recursivas primitivas), pero, traducido a nuestro contexto, se puede expresar así. Nosotros no lo demostraremos hoy aquí y nos conformamos con indicar que la idea es razonar inductivamente, a partir de las funciones básicas y simulando en la lógica operaciones como composición y minimización.

¿Cuál es el dominio de la función recursiva parcial $\phi(i) = \phi_{\mathtt{u}}(\langle i.i \rangle)$? Por definición, $\phi_{\mathtt{u}}(\langle i.i \rangle) = \phi_i(i)$, es decir, el dominio de esta función es justamente K. Por la contribución de Gödel indicada, K es definible. Sea $\kappa(x)$ la fórmula que lo define.

La siguiente contribución de Gödel fue codificar las fórmulas de la aritmética como números. Si se hace con cuidado, no es difícil asegurar que se puede hacer de manera que la función v que, a partir del número n, construye el número asociado a la fórmula $\kappa(\mathtt{n})$ sea recursiva total.

Finalmente, consideramos el conjunto W de los números de Gödel que corresponden a fórmulas verdaderas de la aritmética. La función v que acabamos de construir, entonces, cumple lo siguiente: $n \in K \iff v(n) \in W$ para todo n: es la definición de que $\kappa(x)$ define K.

Por tanto, $K \leq_m W$ y deducimos que W no es decidible. No existe un algoritmo que nos resuelva el Entscheidungsproblem. El conjunto W tampoco es enumerable recursivamente: $\overline{K} \leq_m W$ se demuestra de la misma manera que $K \leq_m W$ pero añadiendo una negación a la fórmula construida.

Razonamientos similares a éste, progresivamente más sofisticados, nos permiten justificar afirmaciones como: no todas las fórmulas verdaderas son demostrables; el conjunto de (los números de Gödel de) las fórmulas demostrables es enumerable recursivamente, completo, y equivalente a K en un sentido muy estricto; la aritmética tiene una fórmula que expresa su propia consistencia, pero, si es cierta, no es demostrable; o el teorema de Tarski de indefinibilidad de la verdad: el conjunto W no es definible en la aritmética.

Es posible argumentar que el conjunto de los índices de funciones totales es "estrictamente más indecidible" que K. Mediante parametrización, podemos obtener una función recursiva total f tal que:

$$\phi_{f(i)}(\langle y.t \rangle) = \begin{cases} 1 & \text{si } \neg T(i,i,y,t) \\ \uparrow & \text{en otro caso.} \end{cases}$$

Entonces, tenemos que si $i \in K$, habrá un par $\langle y.t \rangle$ en que esta función queda indefinida, pero, si $i \notin K$, $\phi_{f(i)}$ es total. Por tanto, esta f nos indica que $\overline{K} \leq_m \{i \mid \forall x\, \phi_i(x)\downarrow\}$ (el conjunto de

índices de funciones totales). Por tanto, $\{i \mid \forall x\, \phi_i(x)\downarrow\}$ no sólo no es decidible sino que tampoco es enumerable recursivamente.

De hecho, es posible construir conjuntos indecidibles "progresivamente más indecidibles": una secuencia infinita de problemas, cada uno "más indecidible" que el anterior, con el primer escalón en K y otros problemas equivalentes a él, como segundo paso $\{i \mid \forall x\, \phi_i(x)\downarrow\}$ y otros equivalentes a él, etc. Y el conjunto de los índices de fórmulas verdaderas, W, resulta ser... ¡"más indecidible" que todos ellos! y, por tanto, "infinitamente más indecidible" que K.