CRÍTICA, Revista Hispanoamericana de Filosofía. Vol. 37, No. 111 (diciembre 2005): 3–38

CONCEPTUALISMO REALISTA Y COMPUTABILIDAD

MAX A. FREUND
Departamento de Filosofía
Universidad Nacional de Costa Rica
mfreund@cariari.ucr.ac.cr

RESUMEN: El artículo formula una interpretación de la computabilidad desde la perspectiva del conceptualismo realista. En esta interpretación, la noción central es la de concepto computable, el cual se entiende como cierto tipo de capacidad cognitiva. Aquí se muestra cómo difiere esa interpretación conceptualista de la clásica, denominada teoría de la computabilidad efectiva, en la cual el concepto fundamental es el de algoritmo; también se discute la relación entre estas dos interpretaciones. La discusión explora las consecuencias de la idea de que los contenidos de los conceptos computables se pueden expresar en algoritmos. En un apéndice se plantean, muy brevemente, las diferencias entre el enfoque conceptualista de la computabilidad y otros dos enfoques no clásicos: el intuicionista y el representacionista.

PALABRAS CLAVE: computabilidad, conceptos computables, conceptualismo realista, algoritmos, computabilidad efectiva

SUMMARY: This paper formulates an interpretation of computability from the perspective of realist conceptualism. The key notion within this theory is that of a computable concept, where a concept is a cognitive capacity of a certain sort. Differences and connections between the conceptualist interpretation and the classical one, customarily referred to as the theory of effective computability, for which the key concept is that of an algorithm, are shown in the text. Consequences of the idea that the content of computable concepts can be expressed in algorithms are explored. An appendix briefly outlines how the present conceptualist approach differs from two other non-classical interpretations of computability, the intuitionist and the representationist.

KEY WORDS: computability, computable concepts, conceptual realism, algorithms, effective computability

El conceptualismo realista es una teoría lógico-filosófica que postula los conceptos como los fundamentos semánticos de todas las expresiones lingüísticas. Se diferencia de otras formas de conceptualismo por su concepción de lo que son los conceptos y de cómo se forman: interpreta los conceptos como capacidades cognitivas o estructuras cognitivas fundamentadas sobre estas mismas capacidades y ubica su formación en un proceso constituido por etapas. Queda claro que, según esta concepción, los conceptos no son entidades *ontológicamente* independientes, sino, más bien, dependientes de seres con disposiciones cognitivas; y son de este modo entidades subjetivas. El realismo de esta versión de conceptualismo se ha de encontrar, más bien, en su interpretación de los predicados nominalizados (como rojez y

humanidad), ya que postula la posible existencia de objetos como denotación para esos términos singulares. Tales objetos son entidades correlacionadas con las condiciones de verdad determinadas por el ejercicio de los conceptos (representados por los predicados). Ahora bien, aunque los conceptos son entidades subjetivas, esto no significa que no sean intersubjetivas. Así como es posible afirmar que varias personas tienen las mismas capacidades, también podemos afirmar que tienen los mismos conceptos.

En Freund 1996, formulé una interpretación del concepto de computabilidad desde el punto de vista del conceptualismo realista, pero no discutí en detalle la relación entre esa interpretación y la interpretación clásica de la computabilidad, conocida como *la noción intuitiva de la computabilidad efectiva*. Esta última noción ha constituido la motivación para desarrollar ciertas teorías formales, como la de recursividad general, la computabilidad Turing, la definibilidad lambda, la computabilidad en el sentido de Post, la computabilidad en términos de algoritmos de Markov, y la computabilidad mediante máquinas de registros ilimitados de Sheperdson y Sturgis.[1] Estas teorías pueden verse como intentos que pretenden capturar en forma precisa la noción de computabilidad efectiva, o bien determinar las características esenciales de las entidades a las cuales hace referencia esa noción, dependiendo de cómo se conciba la llamada tesis de Church: como una reconstrucción racional (en el sentido de Hempel-Carnap) o como una hipótesis (empírica o matemática).

En el presente artículo mostraré las diferencias y las conexiones entre la computabilidad efectiva y la visión conceptualista realista de la computabilidad. El primer concepto se caracterizará, fundamentalmente, tomando como base las concepciones de Alonzo Church y Alan Turing y, en menor grado, las de Post y Gödel. Esto obedece a que los elementos implícitos en caracterizaciones posteriores de computabilidad efectiva ya están comprendidos en las conceptualizaciones de esos cuatro autores.

Las diversas concepciones, ya mencionadas, serán expuestas en la primera sección del artículo, y la caracterización de la computabilidad efectiva en la segunda. En la cuarta sección plantearé mi teoría conceptualista de la computabilidad, la cual contrastaré, en la quinta sección, con la de la computabilidad efectiva. Teniendo en mente la

[1] Para desarrollos amplios de la computabilidad en términos de registros ilimitados, *cfr*. Cutland 1980; en términos de máquinas de Turing, Davis 1982a; de recursividad general, Kleene 1952. Véase también un enfoque menos formalizado basado en la tesis de Church en Rogers 1967.

posibilidad de que no todos los lectores estén familiarizados con el conceptualismo realista, he reservado la tercera sección para hacer una breve exposición general de dicha teoría.

Aparte de la interpretación clásica de la computabilidad, existen otros dos enfoques que han sido presentados como alternativos: el *intuicionista* y el *presentacionista*. Este último considera la distinción entre funciones (en extensión) y descripciones de éstas en un lenguaje interpretado, las cuales son denominadas *presentaciones* de funciones, y sobre esta base formula una teoría intuitiva de la computabilidad, con diferencias importantes de la concepción de la computabilidad efectiva. La teoría intuicionista es una aplicación de los presupuestos constructivos aceptados por filósofos intuicionistas de las matemáticas al concepto de computabilidad. En un apéndice a este trabajo mostraré muy brevemente cómo se diferencian esos dos enfoques de la concepción conceptualista.

1. *Computabilidad efectiva: fundamentos históricos*

Como tema de reflexión, la noción de computabilidad efectiva fue introducida por quienes desarrollaron, originalmente, conceptos formales de computabilidad. Me refiero, fundamentalmente, a Alonzo Church, Alan Turing, Emil Post y Kurt Gödel. Las caracterizaciones de estos autores, sobre todo las de Church y Turing, forman la base de conceptualizaciones de computabilidad efectiva posteriores, las cuales no difieren (en lo esencial) de las primeras.[2] Por esta razón, en la presente sección me ocuparé sólo de los enfoques de esos cuatro autores (con mayor énfasis en los de Church y Turing), con el fin de identificar los elementos en que convergen y a partir de ello, caracterizar la noción de computabilidad efectiva. Dicho de otra manera, la caracterización de computabilidad efectiva que expresaré en este trabajo estará fundamentada históricamente.

El contexto de problemas lógico-matemáticos en el cual Alonzo Church se desenvolvió es bien conocido y, por ello, no voy a ahondar en este tema. Me concentraré, más bien, en sus ideas con respecto a la computabilidad. Church reconoció dos formas de entender la noción de computabilidad efectiva: en términos de aplicación de

[2] Otras caracterizaciones de la noción de computabilidad efectiva pueden encontrarse en Cutland 1980, Kleene 1967, Rogers 1967, Enderton 1977, Davis 1982a y Hermes 1969.

algoritmos o en términos de demostraciones en un sistema formal con determinadas características:[3]

> Así, queda demostrado que no se puede obtener una definición más general de la calculabilidad efectiva que la antes propuesta mediante cualquiera de los dos métodos que surgen de manera natural: (1) definir una función como calculable en forma efectiva si existe un algoritmo para el cálculo de sus valores; (2) definir una función F (para un entero positivo) como calculable en forma efectiva si, para todo entero positivo m, existe un entero positivo n tal que $F(m) = n$ es un teorema que puede ser probado.[4] (Church 1936, p. 58)

En cualquiera de estas dos formas, el cómputo de una función se lleva a cabo en pasos: en la primera forma, el cómputo efectivo de una función supone la aplicación de un método (*el algoritmo*) que producirá una serie de pasos, cada uno de los cuales dependerá de los pasos anteriores:

> Por ejemplo, en el caso de una función F de un entero positivo, un algoritmo consiste en un método por medio del cual, dado cualquier entero positivo n, se puede obtener una secuencia de expresiones (en alguna notación) E_{n1}, $E_{n2}, \ldots, E_{nrn}$, donde E_{n1} es calculable en forma efectiva cuando n es dada; E_{ni} es calculable en forma efectiva cuando n y las expresiones E_{nj}, $j < i$ son dadas, y cuando n y todas las expresiones E_{nj}, hasta E_{nrn} inclusive, son dadas; el hecho de que el algoritmo haya terminado es conocido en forma efectiva y el valor de $F(n)$ es calculable en forma efectiva. (Church 1936, p. 56)

El cómputo en la segunda forma sería una secuencia de pruebas, cada una de las cuales supondría las anteriores.

Es evidente, entonces, que, en la concepción de Church, cada uno de los pasos de un cómputo efectivo será a su vez efectivo; y esto es así porque las reglas que se han de utilizar (en cada uno de los pasos de tales cómputos) tendrán que ser ellas mismas efectivas. Church rechaza totalmente la concepción de un cómputo efectivo que incluya

[3] Entre otras razones, introdujo la segunda forma por el resultado de Gödel de que la noción de recursividad general es absoluta y de que las demostraciones en el sistema involucrado en este resultado son recursivas. Para detalles sobre este tema, véase Gödel 1946.

[4] Las traducciones de los textos citados son mías, a menos que se indique de otro modo.

procedimientos no efectivos.[5] Por ejemplo, en relación con las reglas utilizadas en un cómputo en un sistema formal, Church sostiene:

> Si el sistema ha de servir para todos los propósitos para los cuales se intenta que sirva un sistema de lógica simbólica, es necesario que cada regla de procedimiento sea una operación calculable en forma efectiva, que el conjunto completo de reglas (si es infinito) sea numerable en forma efectiva, que el conjunto completo de axiomas formales (si es infinito) sea numerable de forma efectiva y que la relación entre enteros positivos y la expresión que representa sea determinable en forma efectiva. (Church 1936, p. 57)

En suma, un cómputo efectivo involucrará reglas efectivas (ya sean componentes de un método o de un sistema formal) y una serie de pasos (efectivos) que permita calcular el valor de una función para un argumento dado. Church intentará interpretar la noción de computabilidad efectiva aquí presupuesta en términos de los conceptos formales de recursividad y definibilidad lambda (λ): "Definimos ahora la noción, ya discutida, de una función de enteros positivos *calculable en forma efectiva* identificándola con la noción de una función recursiva de enteros positivos (o de una función de enteros positivos λ-definible)" (Church 1936, p. 56).

Ésta es la formulación original de la llamada *tesis de Church*,[6] la cual se presenta en forma de definición y como una especie de reconstrucción racional (en el sentido de Carnap y Hempel) de la noción de computabilidad efectiva. Como quedará claro más adelante, ésta no es la única forma en que se puede conceptualizar la tesis.

Es importante señalar que, en los primeros años de la década de 1930, Church también concibió su tesis sólo en términos de definibilidad lambda;[7] posteriormente, le dio más peso a la noción de recursividad general (cuya posible identificación con la noción de computabilidad efectiva había sido propuesta por Gödel), a pesar

[5] Estas ideas se expresan claramente en el mismo Church 1936. Para una discusión más amplia de este tema, véanse Sieg 1997 y Gandy 1988.

[6] En la formulación de la tesis en Church 1935 sólo se utiliza el concepto de recursividad general.

[7] En la nota 18 a pie de página de Church 1936, dice: "La cuestión de la relación entre calculabilidad efectiva y recursividad (que aquí pretendo resolver identificando las dos nociones) fue planteada por Gödel en una conversación con el autor. El problema correspondiente de la relación entre calculabilidad efectiva y definibilidad λ había sido propuesta previamente por el autor de manera independiente."

de que para esa misma época ya había sido probada la equivalencia entre definibilidad lambda y recursividad general.[8] Su ulterior preferencia por la recursividad se debe, fundamentalmente, a la posibilidad de probar la tesis mediante el llamado teorema normal de Kleene, interpretando como recursivos los pasos resultantes de la aplicación de un procedimiento efectivo; esto es, como el resultado de aplicar reglas recursivas. Church tenía la idea de que no podía existir una función efectiva cuyo cálculo incluyera reglas no recursivas.[9]

Vemos, entonces, que Church se concentró en los pasos de un cómputo efectivo y en las reglas de las cuales dependen tales pasos para ser llevados a cabo. Tanto los pasos como las reglas tienen que ser efectivos y esto significó, para él, que eran recursivos. Lo anterior permitiría, como ya lo he hecho notar, la justificación de la tesis de Church; esto es, que el concepto de computabilidad efectiva podría ser comprendido en términos de recursividad general. Sin embargo, he de indicar que Church no fundamentó el supuesto de que los

[8] El que dé más importancia a la noción de recursividad es mucho más claro en Church 1935, donde afirma: "En este artículo adopto una definición de la *función recursiva de enteros positivos* que es en esencia de Gödel. Sostengo que la noción de una función efectivamente calculable de enteros positivos debería identificarse con la de función recursiva, pues otras definiciones plausibles de calculabilidad efectiva resultan dar pie a nociones equivalentes a la de recursividad, o más débiles que ésta."

[9] Esta posición es muy clara en Church 1936 y también en una carta dirigida al lógico polaco Józef Pepis. La carta aparece reproducida y comentada en Sieg 1997; en ella se describe el proyecto de Pepis de construir una función numérica computablemente efectiva, pero no recursiva general. Church explica por qué se siente "extremadamente escéptico" con respecto a la posibilidad de existencia de tal función. De acuerdo con Church, una condición mínima para que una *función* f sea computable en forma efectiva es que, para todo entero positivo a, debe existir un entero positivo b tal que $f(a) = b$ tenga una prueba en la matemática. Como toda la matemática conocida hasta ese momento había sido formalizada en sistemas como *Principia Mathematica* (PM) o en alguna de sus extensiones, habría de esperarse una prueba formal, correspondiente a la prueba matemática intuitiva. Pero si f no es recursiva general, por las consideraciones de Church 1936, para cualquier definición de f en el lenguaje de PM, existiría un entero positivo a con respecto al cual no existe un b tal que $f(a) = b$ tenga una prueba en PM o en cualquiera de las extensiones conocidas. Habría que encontrar un principio o regla lógica que no hubiera sido formulada o usada de hecho en la práctica matemática y este principio "debería ser de un tipo tan extraño y, presumiblemente, complicado, que su expresión metamatemática como una regla de inferencia no sería recursiva general" y tendríamos que escudriñar "la alegada aplicabilidad efectiva del principio con considerable cuidado".

pasos de un cómputo efectivo tenían que ser recursivos y esto hace que la justificación de Church, de su tesis, sea arbitraria.[10]

A diferencia de Church, Kurt Gödel no vio, en los conceptos de definibilidad lambda o recursividad, elementos suficientes para interpretar la noción de computabilidad efectiva. Esto lo sabemos por una carta de Alonzo Church, en la cual escribe:

> [Gödel] consideraba totalmente insatisfactoria mi propuesta de que la definibilidad lambda pudiera ser tomada como una definición [de calculabilidad efectiva] [. . .]; más tarde se le ocurrió [a Gödel] que la definición de Herbrand de recursividad, la cual no toma en cuenta la calculabilidad en forma efectiva, podría ser modificada en dirección de la calculabilidad efectiva e hizo esta propuesta en sus conferencias. En ese momento planteó, específicamente, la cuestión de la conexión entre la recursividad en este nuevo sentido y la calculabilidad efectiva; pero dijo que no pensaba que las dos ideas pudieran ser identificadas satisfactoriamente, excepto desde un punto de vista heurístico. (Davis 1982b, p. 9)

También aquí es evidente que, para Gödel, una interpretación de computabilidad efectiva en términos de recursividad tendría que ser tomada solamente como un recurso heurístico. Esta misma concepción se expresa en Gödel 1934, donde, refiriéndose a las funciones recursivas primitivas, el autor afirma:

> tienen la importante propiedad de que, para todo conjunto de valores dado de los argumentos, el valor de la función puede ser computado por un procedimiento finito [. . .]. La inversa parece ser verdadera si, además de recursiones [primitivas], [. . .] se admiten recursiones de otras formas (*v.g.*, con respecto a dos variables simultáneas). Ésta no puede ser probada, puesto que la noción de cómputo finito no está definida, pero sirve como un principio heurístico. (Gödel 1934, pp. 43–44 y n. 3)

Por otra parte, sabemos, por la misma carta de Church, que para Gödel sólo después de un análisis más profundo del concepto de computabilidad efectiva, donde se postularan axiomas relativos a ese concepto, podría hablarse de aceptar, o no, algo como la tesis de Church:[11] "Su única idea, en aquel tiempo, era que, en términos de la calculabilidad efectiva como una noción no definida, podría

[10] Para una discusión sobre este tema, véanse Sieg 1997, Soare 1996, Gandy 1988.
[11] Para detalles sobre este tema, véase Davis 1982b.

ser posible enunciar un conjunto de axiomas que incorporasen las propiedades aceptadas de esta noción, y hacer algo sobre esta base" (Davis 1982b, p. 9).

Así, para Gödel era necesario profundizar más en la noción de computabilidad efectiva, antes de identificarla con algún concepto formal de computabilidad. Los conceptos de recursividad general y definibilidad lambda no contenían elementos suficientes para hacer plausible una interpretación de la noción de computabilidad efectiva. Y es en el análisis de computabilidad de Alan Turing donde Gödel encuentra la profundización requerida.[12] Por su parte, Church (después de la formulación de sus tesis) concuerda con la apreciación de Gödel. Para él, el análisis de Turing establece la conexión de computabilidad efectiva con conceptos formales de manera más patente: "la computabilidad por una máquina de Turing [...] tiene la ventaja de hacer de inmediato evidente la identificación con efectividad en el sentido ordinario (no definido en forma explícita)" (Church 1937, p. 43).

De este modo, los autores de dos caracterizaciones formales del concepto de computabilidad, Gödel con el de recursividad general y Church con el de definibilidad lambda, coinciden en que las ideas de Turing sobre computabilidad proporcionan la comprensión adecuada del concepto de computabilidad efectiva.

El análisis de Turing se basa en una caracterización inicial de lo que es un agente computacional humano ideal (por brevedad, *el agente ideal*),[13] en el cual confluyen los conceptos involucrados en la concepción intuitiva de lo que es una *función producida por un procedimiento mecánico*.[14] El agente ideal de Turing se concibe como un individuo con papel y lápices (en cantidades ilimitadas), asociado a cierto conjunto finito de estados mentales y a un conjunto finito de símbolos. El procedimiento del cálculo del valor de una función, por parte del agente ideal, se lleva a cabo escribiendo los símbolos en un papel, estructurado en forma unidimensional y divido en cuadrados. El agente es capaz de observar, a la vez, sólo un número finito de estos cuadrados y, también, de efectuar un número finito de ope-

[12] Véase Gödel 1946, p. 72n, y 1995, p. 168. En esta última obra dice, refiriéndose a la caracterización de Turing: "Turing estableció, más allá de toda duda, que ésta es realmente la definición correcta de computabilidad mecánica."

[13] Turing llama "computer" al agente ideal, pero es evidente, por sus explicaciones, que no se está refiriendo a una máquina, sino más bien a un ser humano en abstracto.

[14] La estructura del análisis de Turing ha sido desarrollada a fondo en Sieg 1994, 1995, y Gandy 1988.

raciones atómicas o elementales. Éstas son operaciones tan simples que no sería fácil imaginárselas subdivididas en otras operaciones, y su ejercicio dependerá de los estados mentales del agente ideal, así como de los símbolos escritos en los cuadrados observados. Las operaciones atómicas constituyen cambios de símbolos en los cuadrados observados, movimientos a diversos cuadrados observados, pero a cierta distancia del cuadrado observado por el agente ideal. Turing también impuso la condición determinista de que a lo sumo se podía efectuar una operación atómica a partir de un estado mental y un símbolo observado.

Luego de que Turing caracteriza al agente ideal y a lo que se ha dado en llamar "máquinas de Turing",[15] prueba de manera rigurosa la siguiente proposición: *toda función calculable por el agente ideal es calculable por una máquina de Turing*. De aquí procede a aseverar que toda función computable en forma efectiva es computable por el agente ideal y, por lo tanto, es computable por una máquina de Turing, esto es:

(T) Toda función computable es calculable por una máquina de Turing

Esta tesis, llamada *la tesis de Turing*, es más general que la de Church, pues las funciones calculables aludidas por Turing no son, solamente, funciones numéricas:

> Usaremos la expresión "función computable" para designar una función calculable por una máquina, y dejaremos "efectivamente calculable" para referirnos a la idea intuitiva sin identificarla con alguna de estas definiciones en particular. No restringimos los valores que toma una función computable a números naturales; podríamos, por ejemplo, tener funciones proposicionales computables. (Turing 1939, p. 166)

Sin embargo, esto no significa que la noción de computabilidad involucrada en la tesis de Church sólo tenga relación con funciones numéricas. Lo que sucede es que la conexión con funciones no nu-

[15] Esto es, una máquina de estados finitos con una cinta infinita en dos direcciones (dividida en cuadrados, los cuales pueden contener símbolos tomados de un alfabeto finito), con una cabeza que puede leer y escribir esos símbolos, así como examinar el contenido de los cuadrados, uno a la vez. También existe un conjunto de instrucciones (*i.e.*, un programa de Turing), al cual reacciona la máquina en cuestión.

méricas es, en este caso, indirecta.[16] Por otra parte, la restricción de T a funciones numéricas es equivalente a la tesis de Church.[17]

Es evidente, por lo tanto, que Turing pensó los procesos computacionales en términos de aplicación y/o repetición de operaciones elementales sobre símbolos (sean o no matemáticos). Un agente que a todas luces es un ser humano en abstracto llevaría a cabo estas operaciones elementales. Turing no hizo alusión en ningún momento a máquinas calculadoras, y las llamadas máquinas de Turing no son sino una codificación de su análisis de los cálculos efectuados por seres humanos: "Comparamos un hombre en el proceso de cómputo de un número con una máquina [de Turing]"[18] (Turing 1936, p. 231).

Es importante hacer notar que, de modo independiente, Emil Post llevó a cabo un desarrollo formal análogo al de Turing. En

[16] Por ejemplo, se establece una correlación de funciones no numéricas con funciones numéricas de la siguiente forma: se pueden distinguir dos tipos de problemas computacionales, problemas de respuesta afirmativa-negativa y problemas de exhibición de objetos. En el primer tipo de problemas se exige como respuesta un sí o un no; por ejemplo, en cualquier caso de sustitución del esquema de problemas "¿Es la secuencia x de fórmulas bien formadas del sistema del cálculo proposicional Russell-Whitehead una prueba en este cálculo?", o del esquema "¿Es el objeto x una fórmula bien formada del sistema modal S5?" Cada uno de estos esquemas de problemas determina un conjunto de problemas. Consideremos los conjuntos que pueden ser enumerados. Éstos determinarían una secuencia $P_1, \ldots, P_n, \ldots$ numerablemente infinita de problemas cuya respuesta es un sí o un no. Cada uno de los miembros de la secuencia se obtiene asignando un valor al o a los parámetros del esquema de problemas que determina el conjunto. Sobre la base de la secuencia, podemos definir un predicado numérico $A(n) \equiv$ {la respuesta a P_n es afirmativa}. De este modo, el conjunto infinito de problemas se convierte en un predicado numérico. Los problemas de la exhibición de un objeto son aquellos donde se pide un objeto determinado como solución al problema. Por ejemplo, ¿cuál es la fórmula resultante de aplicar *modus ponens* a las fórmulas "$A \Rightarrow B$", "$(A \Rightarrow B) \Rightarrow C$"? Si vemos A, B, C como parámetros, obtendremos un esquema de problemas que da a lugar a una clase infinitamente numerable de problemas $P_1, \ldots, P_n, \ldots$, cuyas respuestas se constituyen en un conjunto infinito de objetos $o_1, \ldots, o_n \ldots$ Sobre la base de estas dos secuencias podemos definir una función numérica $f(n) = b$, si la respuesta a P_n es b.

[17] La equivalencia extensional de la restricción de T a funciones numéricas con la tesis de Church se debe, por un lado, a la prueba de Turing de que toda función λ-definible es T-computable y viceversa, y a la equivalencia extensional de computabilidad lambda con recursividad general, mostrada por Church y S.C. Kleene. (Para estos resultados, véanse Kleene 1936a, 1936b, 1935a, 1935b y 1981, y Turing 1936.) Otras concepciones formales de computabilidad como, por ejemplo, la computabilidad en el sentido de Post y la computabilidad mediante máquinas de registros ilimitados, han sido demostradas como equivalentes, *extensionalmente*, a la computabilidad Turing. El lector puede encontrar pruebas de estas equivalencias en Davis 1982a y en Cutland 1980.

[18] Sobre este tema, véase Gandy 1988, p. 82.

la formalización de Post, existe una cantidad infinita de espacios en ambas direcciones y en los cuales un agente puede trabajar uno a la vez. Es posible que cada espacio esté vacío o tenga una sola marca en él. El individuo que trabaja en los espacios puede llevar a cabo una serie de actos primitivos: marcar un espacio, borrar una marca en un espacio, moverse a la izquierda o a la derecha, determinar si el espacio en que se encuentra está o no marcado. Todo esto, en unión a un conjunto de instrucciones, servirá para solucionar problemas. Ese conjunto de instrucciones le indicará al agente cuáles de los actos primitivos puede llevar a cabo y cuándo dejar de efectuarlos. Con su teoría de computabilidad, Post, al igual que Turing, busca abstraer y caracterizar aspectos de las capacidades computacionales de los seres humanos.[19] Es más, la tesis de Church constituye, para él, una hipótesis empírica sobre esas mismas capacidades y critica a Church por ver su tesis como una definición, pues "esconde el hecho de que se ha logrado un descubrimiento fundamental en las limitaciones del poder matematizante del *Homo sapiens* y nos hace insensibles a la necesidad de su verificación continua" (Post 1936, p. 291n). Para Post, la idea es que el trabajo de investigación en computabilidad conduzca a que la tesis de Church deje de ser una hipótesis para convertirse en una ley natural.[20]

2. *Computabilidad efectiva: caracterización*

Lo expuesto en la sección anterior proporciona, claramente, bases suficientes para construir una caracterización del concepto clásico de computabilidad; esto es, del concepto de computabilidad efectiva. Para este fin, asumiré la tesis Turing-Church, aun tomando en cuenta que ha sido algunas veces cuestionada.[21] Y la asumiré, fundamentalmente, porque esas objeciones parten de supuestos muy discutibles y la evidencia a favor de la tesis es de aceptación más universal.[22] Por otra parte, aunque el problema de si "la tesis Turing-Church" es una

[19] Sin embargo, para justificar su análisis formal, Post no efectúa un análisis del agente, como sí lo realiza Turing. Tampoco prueba, a diferencia de Turing, la equivalencia extensional de ese análisis a los de Church y Gödel.

[20] "El éxito del programa antes mencionado haría, para nosotros, que esta hipótesis se convirtiera en una ley natural, no en una definición o en un axioma" (Post 1936, p. 291n).

[21] Entre los cuestionamientos más importantes están los de Kálmar 1959, Péter 1959 y Bowie 1973.

[22] Para una crítica de las posiciones de Kálmar y Péter, véase Mendelson 1963, y de la de Bowie, véase Ross 1974.

definición o una tesis (en el sentido propio) es filosóficamente importante, por la temática del presente artículo no profundizaré en esta discusión.[23] Me limitaré a asumir la tesis sin presuponer concepción alguna sobre su fundamento epistemológico.[24]

Así, dada la tesis Turing-Church, la equivalencia extensional de las diferentes interpretaciones formales del concepto de computabilidad efectiva, junto con las ideas del análisis de Turing, nos ha de llevar a un escenario en el que un agente, en cierto contexto, tendrá la capacidad de reaccionar de determinado modo ante un conjunto de instrucciones o reglas con ciertas características. La naturaleza del contexto, las características de las instrucciones y las capacidades del agente, propios de ese escenario, deberán esclarecer lo que ha de entenderse por computabilidad efectiva.

El conjunto de reglas, o instrucciones (el *algoritmo*) al cual reacciona el agente computacional es un conjunto finito, cuyas componentes pueden ser aplicadas en un número discreto de pasos; cada uno de estos pasos sólo requiere un tiempo finito. Además, tales reglas no involucran elementos de azar, recursos analógicos o continuos y su aplicación no requiere el uso del ingenio;[25] pero sí proporcionan un resultado en un tiempo finito. En suma, la reacción del agente al algoritmo es de tal forma que, dado un ingreso de datos, el cómputo se llevará a cabo en la forma de pasos discretos, mecánicamente, sin el uso de recursos analógicos (continuos) o métodos aleatorios. De este modo, los pasos de un cómputo efectivo serán ellos mismos efectivos, pues cada paso genera un resultado producto de la aplicación de reglas, que es puramente mecánica.

El contexto en el que se ubica el agente es una situación ideal en la que es posible aplicar el algoritmo. En esta situación se encuentran todos los elementos necesarios para llevar a cabo, guardar y recuperar los pasos de un cómputo. De esos elementos, es importante resaltar las coordenadas tiempo y espacio en que el agente se desarrollará: se asume que el agente no estará limitado por el tiempo requerido para ejecutar las instrucciones del algoritmo ni por los recursos espaciales mínimos para almacenar los resultados de la aplicación de cada una de esas instrucciones.

[23] Véase una discusión sobre este tema en Shapiro 1981.

[24] A manera de ejemplo cabe citar el problema de si la tesis puede ser probada matemáticamente. Debido a la vaguedad de los conceptos involucrados en la noción de computabilidad efectiva, se suele mantener la idea de que la respuesta a ese problema es negativa; sin embargo, véase una posición contraria en Mendelson 1990.

[25] Esta noción de ingenio es, por supuesto, problemática, pero su contenido se dará por sentado en forma intuitiva en este trabajo.

Las capacidades del agente son de una naturaleza tal que le permitirán percibir símbolos, identificar, a través del cómputo, si son los mismos o si difieren. Posee también capacidades que le permiten implementar instrucciones, tal como la transformación de símbolos; asimismo cuenta con la capacidad para entender el lenguaje de las instrucciones. Después de todo, el agente es un ser humano en abstracto y no habría razón para suponer que esa capacidad queda anulada en un proceso de cómputo. Y es que se podría asumir lo contrario; esto es, se podría suponer que el agente sigue las instrucciones de forma implícita. Dicho de otra manera, el agente seguiría las instrucciones del algoritmo no porque las hubiera leído y comprendido, sino más bien porque sus reacciones en un proceso computacional, en las que se aplican (de forma inmediata) operaciones a datos contemplados, son fruto de un determinismo interno en el agente que lo hace seguir ciertas instrucciones. Es decir, tales reacciones no se ven como el producto de una comprensión de instrucciones, sino como resultado de una reacción mecánica meramente causal. Pero éste no parece ser el enfoque de los autores que aquí nos ocupan; por ejemplo, en una de las dos caracterizaciones del agente ideal, en Turing 1936, se hace una interpretación de los estados mentales del agente en la cual parece suponerse la capacidad para comprender el lenguaje en la acción del cómputo:

> evitamos introducir el "estado mental" al considerar una contrapartida suya más física y definida. Siempre es posible para el agente suspender su trabajo, irse y olvidarse de todo, y luego regresar y continuar con la tarea. Si hace esto, deberá dejar una nota de instrucciones (escritas de alguna forma estándar) explicando cómo ha de continuarse el trabajo. Esta nota es la contrapartida del "estado mental". Supondremos que el agente trabaja de manera tal que no efectúa más de un paso en una sentada. La nota de instrucciones le deberá permitir llevar a cabo un paso y escribir la siguiente nota. Así, el estado de progreso del cómputo en cualquier etapa estará completamente determinado por la nota de instrucciones y los símbolos de la cinta.[26] (1936, p. 253)

Aquí, evidentemente, el agente tiene la capacidad de comprensión lingüística. Asumiré, por lo tanto, que se presupone cierto nivel de comprensión lingüística como capacidad propia del agente.

Ahora bien, los posibles objetos a cuyo cómputo se dedica el agente al reaccionar al algoritmo son varios. Es evidente que los

[26] Traduje como "agente", el término "computer"; la razón de esto se explica en la nota 13.

autores antes considerados ven, en los números, funciones numéricas o funciones proposicionales, entidades cuya computabilidad se puede afirmar. Sin embargo, autores posteriores a los mencionados se centran en la computabilidad de funciones (sean numéricas o no) y establecen la computabilidad de otras entidades a partir de éstas. Así, por ejemplo, la computabilidad de un predicado se determina por la computabilidad de cierta función asociada al predicado y se conoce como "la función característica del predicado".

Por ser la práctica aceptada, adoptaré aquí el enfoque centrado en las funciones, como los objetos fundamentales de la computabilidad efectiva. También atribuiré a la computabilidad efectiva la interpretación extensional de las funciones que esa misma práctica ha aceptado.[27] Esto significa que, como criterio de identidad de dos funciones f y g, se usarán tanto la identidad del rango de argumentos como la identidad de valores para cada argumento; esto es, para todo elemento α del rango, $f(\alpha)$ es igual a $g(\alpha)$. Como es bien sabido, ésta no es la única manera de concebir las funciones; también es posible hacer una interpretación intensional. Así, Church afirma:

> Es posible [...] permitir que dos funciones sean diferentes sobre la base de que la regla de correspondencia difiere en significado en los dos casos, aunque siempre se produzca el mismo resultado cuando se aplican a cualquier argumento. Cuando se hace esto decimos que estamos tratando con *funciones en intensión*. La noción de diferencia de significado entre dos reglas de correspondencia es vaga, pero en términos de algún sistema de notación, puede formularse con exactitud de diversos modos. (1941, p. 3)

Sería importante emprender una discusión de la interpretación conceptualista de la computabilidad en relación con una interpretación clásica que asuma el concepto intensional de función; sin embargo, no lo haré aquí (dejaré la tarea para otro trabajo).

3. *Conceptualismo realista*

Habiendo expuesto ya una caracterización de computabilidad efectiva, ahora es posible plantear mi teoría conceptualista de la computabilidad; no obstante, dejaré la presentación para la cuarta sección,

[27] Por ejemplo, véanse Davis 1982a, Rogers 1967, Epstein y Carnielli 2000 y Cutland 1980.

pues en la presente expondré, a grandes rasgos, algunos de los aspectos más importantes del conceptualismo realista. Esta exposición está dedicada, fundamentalmente, a los lectores no familiarizados con esa teoría filosófica; por obvias limitaciones, la exposición será muy breve.

El conceptualismo realista es, por una parte, una teoría filosófica de la atribución y de predicados nominalizados, la cual (a diferencia del nominalismo y el realismo lógico) postula conceptos como fundamentos semánticos para la aplicación correcta o incorrecta de los predicados. Como ya lo señalé en la introducción, el conceptualismo realista entiende los conceptos como capacidades cognitivas o como estructuras cognitivas basadas en las primeras, realizables intersubjetivamente. En el caso de los predicados, los conceptos que éstos representan (llamados *conceptos predicables*) son las capacidades cognitivas cuyo ejercicio permite clasificar o relacionar objetos de varios modos. Los conceptos predicables corresponden a los rasgos del pensamiento que determinan las condiciones de verdad de los predicados en diferentes contextos de uso y, de esta forma, son capacidades cognitivas que subyacen en nuestras habilidades para seguir reglas en el uso correcto de los predicados.

Ahora bien, los predicados no son los únicos elementos lingüísticos considerados por el conceptualismo realista. De hecho, esta teoría postula conceptos para todas las expresiones lingüísticas significativas, componentes de las oraciones; por ello constituye una teoría semántica filosófica general. Como su método incluye la construcción de un sistema lógico-formal, en el cual se llevan a cabo las interpretaciones de las expresiones del lenguaje natural, la teoría intenta convertirse en una semántica lógica para lenguajes naturales.[28]

Los diversos conceptos que representan las expresiones lingüísticas se forman, según el conceptualismo realista, en un proceso constituido por diferentes etapas. En tal proceso, las estructuras cognitivas desarrolladas en cualquiera de las etapas no son, en general, definibles explícitamente en términos de las formadas en las etapas previas ni tampoco son reducibles a ellas. En ninguna etapa de ese proceso se desecha lo logrado en etapas previas; los logros se conservan, más bien, como partes útiles de todo el marco conceptual.

Hay dos formas posibles, no necesariamente incompatibles, de llevar a cabo el proceso de construcción de conceptos, las cuales

[28] Para una descripción más detallada del conceptualismo realista como proyecto de una semántica lógica para lenguajes formales, véase Freund 2002.

corresponden a dos variantes del conceptualismo realista. El conceptualismo holístico —la variante más amplia— supone que el proceso ha de llegar a una fase en la que toda expresión —supuestamente significativa— componente de una oración, de un lenguaje dado, representará un concepto, y donde es posible la construcción impredicativa de conceptos; esto es, la formación de conceptos cuyo contenido presupone la totalidad a la cual pertenecen. La otra perspectiva de cómo ha de evolucionar la formación de conceptos es la del conceptualismo constructivo. A diferencia del conceptualismo holístico, esta variante filosófica sostiene que sólo pueden formarse conceptos predicativos; es decir, conceptos que respetan el principio Poincaré-Russell del círculo vicioso.

A pesar de que la construcción constructivista es muy restrictiva, el conceptualismo holístico la presupone, pues asume que la construcción predicativa de conceptos es la base que conduce al proceso impredicativo de formación de conceptos. Existe un mecanismo cognitivo que nos lleva de las etapas predicativas de elaboración de conceptos a las impredicativas. Por otra parte, es posible que el conceptualismo constructivo no acepte el enfoque holístico. Una forma estricta de esa variante de conceptualismo rechaza de plano la formación impredicativa por considerarla cognitivamente imposible.

Como ya lo señalé, los conceptos predicables constituyen la base semántica de los predicados. Ahora bien, complementarios a los conceptos predicables, existen otros que se forman en el proceso ya mencionado, cuya naturaleza es tal que, cuando uno de ellos es ejercido con un concepto predicable en un acto mental o del habla, ese concepto es el elemento que imprime al acto una naturaleza referencial. Por esta razón, a tales conceptos se los llama "conceptos referenciales". En suma, los conceptos referenciales dan cuenta de la referencia a objetos en un acto mental o del habla, y los predicables de la naturaleza predicable de ese mismo acto.

Muchos de los conceptos referenciales se forman sobre la base de lo que llamaré "conceptos sustantivables contables" (*conceptos contables*, por brevedad). Este último tipo de conceptos son capacidades cognitivas intersubjetivas, cuyo uso en el pensamiento y la comunicación está asociado con ciertos criterios de identidad; esto es, criterios mediante los cuales somos capaces de distinguir, contar y clasificar objetos. Expresiones lingüísticas como "vaca", "caballo" e "insecto" representan conceptos de ese tipo. Esta clase de expresiones pueden formar parte de construcciones lingüísticas más complejas, como "alguna vaca", "todo caballo" y "cualquier insecto", es decir, *frases de*

cuantificación relativa, las cuales representan conceptos referenciales.[29] El ejercicio de los conceptos que representan esas frases nos permiten referirnos a objetos de cierto modo.

Ahora bien, no todo concepto referencial involucra un concepto contable. Éste es el caso de los conceptos representados por *frases de cuantificación absoluta*; por ejemplo, las expresiones "cualquier cosa", "alguna cosa", "todo individuo" y "muchas entidades". Los conceptos incluidos en estas frases, como "cosa" e "individuo", no son contables, debido a que no incluyen criterios de identidad.

Los conceptos referenciales nunca son parte de lo que imprime, a un acto mental o del habla, una naturaleza predicable. Como ya lo señalé, su papel consiste en imprimirle una naturaleza referencial al acto. En estas circunstancias, se dice que el concepto referencial se ejerce de forma activa. Cuando una frase cuantificacional aparece en el contexto de un predicado complejo, ésta ya no cumple su papel referencial (esto es, no representa un concepto referencial), debido a que el concepto que representa el predicado complejo imprime al acto del habla una naturaleza predicable. En esta situación, se dice que el concepto referencial que representa la frase cuantificacional ha sido *desactivado*. Esto significa que el concepto predicable se forma sobre la base del contenido intensional del concepto referencial, representado por el predicado complejo que contiene la frase cuantificacional. Así, por ejemplo, en el enunciado "Juan busca algún entretenimiento cuando va al cine", la oración componente "Juan busca algún entretenimiento" contiene la frase cuantificacional "algún entretenimiento" como parte del predicado "buscar algún entretenimiento". El conceptualismo realista sostiene que, en estos contextos, cualquiera de esas frases representa el contenido intensional de los conceptos referenciales que tales frases usualmente representan.

Otro aspecto importante en el conceptualismo realista se relaciona directamente con las nominalizaciones de los predicados; esto es, con las transformaciones de predicados en términos singulares. En esta teoría no se asume que toda nominalización denota, pero sí se presupone que, en caso de que una nominalización denote, lo denotado habrá de ser un objeto. Sin embargo, no hay presupuesto alguno con respecto a la naturaleza de tal objeto, aparte del requisito de que debe haber una conexión entre el objeto denotado por el predicado nominalizado y el concepto que tal predicado representa. Como lo indiqué

[29] He desarrollado la lógica de los conceptos contables en Freund 2000, 2001 y 2004.

en la introducción, la conexión debe basarse en una relación del objeto con las condiciones de verdad determinadas por el posible ejercicio del concepto. Por esta razón, las posibles denotaciones de predicados nominalizados son llamadas también correlatos de conceptos. El objeto denotado puede ser intensional, y en este caso hablamos de un conceptualismo realista intensional; si el objeto es extensional, de un conceptualismo realista extensional.

Es necesario mencionar que los nombres propios y las descripciones definidas son interpretadas, en el conceptualismo realista, como casos especiales de sustantivos contables, pues se considera que representan conceptos contables (los cuales proporcionan criterios de identificación de, a lo sumo, un objeto). Es más, de acuerdo con el conceptualismo realista, los nombres propios y las descripciones definidas usadas en el contexto de una oración pueden o no acarrear contenido existencial. Así, en el contexto de la oración "Pegaso es un caballo blanco", el nombre propio "Pegaso" representa un concepto referencial en el cual se asume que el único objeto identificable (por el criterio de identidad provisto por el concepto contable asociado a Pegaso) existe y, también, puede ser usado de tal forma que represente un concepto referencial en el cual no se ha hecho tal supuesto.

El proceso de formación de conceptos, aceptado por el conceptualismo realista, incluye también las nociones lógicas. Entre las primeras nociones lógicas desarrolladas se encuentran los conceptos de identidad relativa a sustantivos contables, tal como el expresado en "*a* es el mismo caballo que *b*". La construcción de las identidades relativas precede a la formación de la identidad absoluta o irrestricta; por ejemplo, la noción de que *un objeto* a *es el mismo que* b. Por cierto, ésta también presupone la construcción previa de la noción de cuantificación sobre conceptos contables, pues se construye como la noción de identidad con respecto a uno u otro concepto contable.

La cuantificación absoluta (esto es, el concepto expresado lingüísticamente, por ejemplo, en las expresiones "todo individuo", "todo objeto" o "toda entidad") también requiere la formación previa de cuantificaciones relativas y cuantificación (de segundo orden) sobre conceptos contables. Sin embargo, el conceptualismo realista deja abierta la posibilidad de que haya una etapa de formación de conceptos en la cual se forme la noción de cuantificación absoluta como referencia a todos los objetos en general (esto es, como referencia a todos los objetos, independientemente de que caigan o no bajo algún concepto contable), aunque presuponga la perspectiva filosóficamente problemática de que existen objetos para los cuales no hay criterios de identidad. Lo mismo se aplica al concepto de identidad absoluta.

Hasta el momento he presentado algunas de las características de los conceptos que el conceptualismo realista postula como base semántica para expresiones lingüísticas. No voy a seguir profundizando en esto, pues el objetivo de la presente sección es meramente instrumental: brindar al lector un panorama general de algunos de los elementos del conceptualismo realista, de tal forma que pueda comprender mejor lo que se discute en las siguientes secciones.

4. *Conceptualismo realista y conceptos computables*

Queda claro, por la anterior sección, que para el conceptualismo realista hay conceptos que constituyen capacidades cognitivas (realizables intersubjetivamente), las cuales permiten clasificar, relacionar o identificar objetos. Ahora distinguiré, en relación con este tipo de conceptos, entre los computables y los no computables. Los primeros están constituidos por los totalmente computables y los semicomputables.[30] Y por un concepto *totalmente computable* entenderé aquel cuyo ejercicio permite determinar, en un tiempo finito, que ciertos objetos *deban* o *no* ser clasificados, identificados o relacionados tal como lo hace el concepto, sin que esto presuponga la existencia de estados cognitivos ligados a prácticas que involucran el azar o el ingenio. Semicomputable será todo concepto cuyo ejercicio permite decidir (en un tiempo finito y sin asumir la existencia de estados cognitivos que involucren prácticas en las que se haga uso de instrumentos de azar o del ingenio) que determinados objetos sean clasificados, identificados o relacionados de la forma presupuesta por el concepto, en caso de que la extensión del concepto abarque efectivamente estos objetos.

Las funciones numéricas como la suma y la multiplicación, tomadas como conceptos, constituyen buenos ejemplos de conceptos computables. Es necesario tener en cuenta que esos conceptos deben ser distinguidos de las funciones en extensión de la computabilidad efectiva; los primeros son entidades intensionales, mientras que las segundas son extensionales. Ahora bien, esto no significa que los conceptos numéricos necesariamente estén desvinculados de las funciones como entidades extensionales. Como señalé en la sección anterior, un aspecto importante del conceptualismo realista está ligado al proceso conocido como *nominalización* de predicados. Si una nominalización denota, lo que denota es un objeto que, de algún

[30] Sería interesante explorar formalmente una teoría de conceptos computables que haga la distinción entre conceptos computables contables y conceptos computables predicables.

modo, está asociado a las condiciones de verdad del concepto que representa el predicado nominalizado, y se le denomina, como ya lo indicamos, el correlato del concepto. Las funciones numéricas de la teoría de Zérmelo-Fraenkel podrían ser usadas como correlatos de los conceptos numéricos.

La caracterización de concepto computable, formulada anteriormente, involucra la noción de ejercicio de un concepto. Aquí no voy a caracterizar ni a definir este concepto; es decir, lo adoptaré como un concepto básico o primitivo (en el sentido lógico). Por la sección anterior, el lector tendrá ya una idea del empleo de esta noción dentro del conceptualismo realista. Los usos de las diversas expresiones lingüísticas, en diversas situaciones, constituyen ejemplos claros del ejercicio de los conceptos, pues éstos constituyen su fundamento semántico. De este modo, las aseveraciones o inferencias que incluyan un predicado —como por ejemplo "rojo"— son ejemplos del ejercicio del concepto de rojo; la expresión lingüística de la suma de varios números manifiesta el ejercicio del concepto de adición. Pero no sólo en actos lingüísticos encontramos situaciones que indican la posesión y el ejercicio de conceptos. Hay situaciones en las que evidentemente existe una categorización o identificación de individuos, sin la previa adquisición del lenguaje, como en el caso de los infantes, o las acciones claramente derivadas de la identificación de individuos sin un acto lingüístico previo (interno o externo) —como cuando Juan huye tras identificar un leopardo sin emitir palabra alguna interna o externamente—; dichas situaciones revelan ejercicio y posesión de conceptos.

Es necesario señalar que el ejercicio de un concepto puede requerir ciertas condiciones internas y, tal vez, condiciones externas al agente que ejercita el concepto. Así, por ejemplo, para que un vidente pueda ejercer el concepto de rojo en la percepción es necesario que pueda ver (condición interna) y que haya cierta cantidad de luz en ese ambiente (condición externa). En el caso del ejercicio del concepto computable de suma puede exigir, en ciertas circunstancias, determinada capacidad de memoria que el agente no posee y que le imposibilita ejercer ese concepto. En estos casos, es posible que el agente conecte cognitivamente el concepto computable con otros conceptos computables que permitan solventar la situación. Así, el agente puede hacer uso de medios externos como papel y lápiz, los cuales, junto con sus capacidades motoras y perceptuales para escribir números, generan un estado cognitivo computacional en el que las limitaciones de su capacidad de memoria queden superadas y, así, pueda ejercer el concepto.

Ahora bien, en el presente enfoque conceptualista de la computabilidad no se presupone que el agente tenga *de facto* todas las condiciones internas y externas que le permitan ejercer cualquier concepto computable que se haya formado. Es una cuestión más de idealidad: si el agente poseyera las condiciones internas y estuviera situado en las condiciones externas que requiere el ejercicio de un concepto computable, el ejercicio de este concepto permitiría determinar en un tiempo finito, sin asumir la existencia de estados cognitivos en que se presuponga el uso de instrumentos de azar o el ejercicio del ingenio, que ciertos objetos *deban* o *no* ser clasificados, identificados o relacionados tal como lo hace el concepto.

En el ejercicio de un concepto computable, la inexistencia de estados cognitivos conectados a prácticas ligadas al azar implica que el ejercicio de este concepto no ha de suponer una condición interna al agente, la cual involucre conceptos cuyo ejercicio lleve a estados cognitivos ligados a esas prácticas. Así, el ejercicio de un concepto computable como el de suma no debería presuponer, por ejemplo, el estado cognitivo generado en situaciones en las que se tira una moneda al aire (esto es, en las cuales se ejercen conceptos ligados a actividades motoras) y se busca percibir cuál de las dos caras de la moneda, ya en el suelo, es visible. El agente recibe del medio dos números y puede ejercer, solamente con esto, su capacidad para sumar. El ejercicio de esta capacidad requerirá cierto tiempo, pero éste siempre será finito: el agente habrá terminado de sumar después de un lapso determinado. Este tiempo podrá variar de acuerdo con las características propias del agente y con la complejidad de los conceptos computables involucrados. El tiempo requerido para aplicar el concepto de suma a 1.350 y 6.470, por ejemplo, podría no ser el mismo que el que se necesita para aplicarlo a 2 y 3; y es posible que el tiempo requerido en la primera suma varíe de un agente a otro. Poder efectuar las sumas tampoco deberá implicar, necesariamente, el uso de estados cognitivos cuya caracterización suponga la noción de ingenio del agente. El ejercicio de un concepto computable tampoco requerirá, necesariamente, el seguimiento de instrucciones. Es más, se puede ver como un proceso continuo que, tal vez, no podría ser representado por una secuencia de pasos.

Por lo tanto, la interpretación conceptualista de la computabilidad sugiere un escenario en el cual se encuentra un agente con múltiples capacidades cognitivas de determinado tipo (a las cuales he llamado "conceptos computables") en ciertas condiciones internas y externas que le permiten ejercer esos conceptos. Así, un problema es *computacionalmente solucionable* (desde el punto de vista conceptualista)

si y sólo si, en principio, el agente logra formarse uno o varios conceptos computables, así como condiciones internas y externas tales que permitan que esos conceptos sean ejercidos y, una vez ejercidos, proporcionen la solución al problema.

5. *Computabilidad efectiva y computabilidad conceptualista*

La noción de computabilidad conceptualista involucra conceptos computables, y la noción de computabilidad efectiva, algoritmos y funciones (en el sentido extensional, esto es, como conjuntos de pares ordenados que cumplen con ciertas características). La existencia de las entidades presupuestas por ambas nociones difiere claramente en puntos importantes. Los conceptos constituyen un conjunto de entidades intensionales que dependen, ontológicamente, de seres finitos capaces de formar conceptos. De ahí que la cardinalidad de ese conjunto no esté del todo clara: en nuestro mundo sería a lo sumo numerable, pues, durante su existencia, cada uno de esos seres sólo es capaz de formar un conjunto finito de conceptos[31] y, dado que la totalidad de esos seres es a lo sumo numerable (debido a las coordenadas espacio-temporales finitas en que se tiene que desarrollar cada uno de ellos), entonces, por un simple argumento de cardinalidad, el conjunto posible de conceptos que pudieran ser formados como totalidad tendría que ser a lo sumo numerable.[32] Sin embargo,

[31] La posibilidad de formar un número infinito de capacidades en un tiempo finito estaría excluida para seres de nuestro universo, dada la finitud de su capacidad de almacenamiento. Existe un número finito de partículas elementales en nuestro universo que imposibilita una infinita capacidad de almacenamiento en cualquier entidad espacio-temporal.

[32] Dado que en la noción de concepto computable hay cierta vaguedad, es posible poner en duda este argumento sobre la idea de que los conjuntos deben estar determinados por condiciones precisas. Sin embargo, este criterio de determinación de conjuntos no se acepta en la práctica de la investigación, la teorización y la publicación en lógica. A modo de ilustración, considérese primero la semántica de mundos posibles. Como es bien conocido, dos conceptos fundamentales en esa semántica son el de mundo posible y el de individuo posible. Se asume que existe un conjunto de mundos posibles y un conjunto de individuos posibles (en este último caso, cuando se distingue entre existencia y existencia posible) y con éstos se construyen los modelos semánticos para muchos de los lenguajes formales intensionales. Ahora bien, en esos dos conceptos hay una evidente oscuridad y vaguedad y, de ahí las múltiples discusiones filosóficas con respecto a ellos, como aquellas en favor de una interpretación realista, nominalista o conceptualista de los mundos posibles, o las que versan sobre el esencialismo y las relativas a la identificación de individuos a través de mundos posibles. Sin embargo, toda esa vaguedad y toda esa oscuridad no han obstado para que lógicos y matemáticos sigan haciendo uso de tales conceptos al determinar los conjuntos base de la semántica de la lógica intensional.

no podemos identificar los conceptos que en principio pueden ser formados con los conceptos que van a ser formados por la totalidad de los seres capaces de formar conceptos, durante su existencia en nuestro universo. Es posible que haya conceptos cuyas condiciones de formación nunca estén dadas en nuestro universo y que sólo habrían podido formarse en mundos físicamente posibles alternativos al nuestro, en los cuales pueden ser dadas esas condiciones y la naturaleza de los seres en cuestión sea igual que la de los seres de nuestro mundo. Si el número de variaciones físicamente posibles fuera a lo sumo numerable, pues también la cardinalidad del conjunto de conceptos posibles sería a lo sumo numerable, por el hecho de que cada ser finito capaz de formar conceptos en un mundo físicamente posible, donde las condiciones permiten la formación de conceptos que no serían formados en nuestro mundo, sólo podría formarse un número finito de conceptos durante su vida. Sin embargo, si el número de variaciones es no numerable, la cardinalidad del conjunto de conceptos posibles podría ser un número transfinito mayor que $\aleph_0$. En esta última situación no podríamos determinar claramente si el número de conceptos computables es o no a lo sumo numerable.[33]

El caso de la naturaleza de las funciones (en extensión) y los algoritmos es distinto. La existencia de un algoritmo y de una función no presupone su construcción en una coordenada espacio-temporal. Las instrucciones que componen el algoritmo son secuencias posibles del

Otro ejemplo se encuentra en la aplicación de la lógica de primer orden, con su semántica *estándar*, a la evaluación de argumentos en el lenguaje natural. Al usarse la semántica de teorías de modelos con este fin, hay que especificar primero el dominio de los cuantificadores, el cual, según se entiende, ha de ser un conjunto. Pero si adoptamos la idea de que los conjuntos deben especificarse de forma precisa, la traducción requerida para tal evaluación sería imposible en el caso de una gran cantidad de argumentos. A modo de ilustración, consideremos el siguiente razonamiento: *Todo ser racional es una persona. Toda persona tiene derecho a la vida. Por consiguiente, todo ser racional tiene derecho a la vida*. Para evaluar este argumento en la lógica de primer orden, primero lo traducimos a un lenguaje formal de primer orden, para lo cual es necesario determinar el dominio de los cuantificadores. Aquí habría tres opciones: (1) el conjunto de las personas; (2) el conjunto de los seres racionales, y (3) el conjunto de los individuos; pero los conceptos de persona, ser racional o individuo no son conceptos precisos. Si se exige precisión en la determinación de conjuntos, no se puede aceptar como conjuntos los descritos en 1–3. Y con esto, no obtendríamos el dominio necesario para traducir el argumento mencionado; sin embargo, los textos de lógica sí avalarían los dominios especificados en 1–3 y, en general, cualquier dominio especificado por conceptos no precisos.

[33] No obstante, más adelante exploraremos la posibilidad de que sea numerable, sobre la base de que los contenidos de estos conceptos son expresables lingüísticamente.

alfabeto de un lenguaje y su existencia consiste en ser una posibilidad lógica. Existe, en el sentido de necesidad lógica, un conjunto a lo sumo numerable de oraciones finitas de un lenguaje con un alfabeto a lo sumo numerable y, por ende, un conjunto a lo sumo numerable de algoritmos formulables en un alfabeto de ese tipo, pues éste estará compuesto por un conjunto finito de instrucciones. Ahora bien, el sentido de computabilidad efectiva no nos indica el tipo de lenguaje aceptable para formular las instrucciones y la forma que éstas deberían tener; no obstante, como se supone que entre los posibles agentes computacionales se encuentra un ser humano abstracto, el cual no podría implementar instrucciones de longitud infinita (estén basadas o no en un alfabeto infinito), tendremos que suponer que los lenguajes de los algoritmos son finitistas en su sintaxis. Además, como ya se mencionó, es un presupuesto de la noción de computabilidad efectiva que los algoritmos estén compuestos por conjuntos finitos de instrucciones. Por lo tanto, sobre el supuesto de que el agente computacional podría ser un ser humano, el número de algoritmos posibles tendría que ser, a lo sumo, numerable. Y como existe un número numerablemente infinito de funciones numéricas constantes, es evidente que el número de algoritmos es infinitamente numerable.

Por otra parte, dada la tesis de Turing-Church, a cualquier función computable por un algoritmo le correspondería un algoritmo de un lenguaje finitista, tal como el de las máquinas de Turing. Como el conjunto de algoritmos del lenguaje Turing es infinitamente numerable así como el conjunto de funciones Turing computables, el número de funciones computables (en general) tiene que poseer esa misma cardinalidad. De este modo, la tesis de Church nos proporciona un elemento adicional para fundamentar la idea de que la cardinalidad del conjunto de funciones computables de manera efectiva es infinitamente numerable, cosa que no ocurre, como hemos visto, con los conceptos computables.

Las nociones de existencia y posibilidad involucradas en los conceptos de computabilidad efectiva y de computabilidad conceptualista son, por lo tanto, diferentes. El tipo de posibilidad ligada a la conceptualista es, obviamente, más restringida que la modalidad implicada en la computabilidad efectiva. Los conceptos computables son capacidades que un ser (en condiciones de formar conceptos) puede desarrollar de acuerdo con factores internos de su estructura biológica y con factores externos que posibilitan el desarrollo de esos conceptos.[34] La posibilidad que aquí interviene no llega a tener la

[34] Como señalé en la sección 4, existen dos formas de conceptualismo que

amplitud de la posibilidad (lógica) presupuesta en la computabilidad efectiva.

Si bien las modalidades involucradas en ambas nociones son diferentes y la cardinalidad de las posibles entidades implicadas podría ser diferente, existe una dependencia cognitiva de los algoritmos en los conceptos computables: los conceptos computables proporcionan la base cognitiva en la aprehensión de un algoritmo como procedimiento para computar una función. Con un ejemplo puedo indicar la razón de esta última afirmación.

Considérense las funciones numéricas computables: los valores de una misma función numérica pueden ser computados por diferentes algoritmos, pero todos estos posibles algoritmos pueden ser asociados a la misma función, porque son aprehendidos cognitivamente como algo que calcula sus valores. Si no fuera por tal aprehensión, los diferentes algoritmos serían sólo conjuntos de instrucciones que, dados los mismos datos de ingreso, producen los mismos datos de salida; esto es, no se establecería una conexión entre los algoritmos y la función. La aprehensión es posible sólo si se ha construido el concepto correspondiente a la función numérica. Por ejemplo, si el concepto computable de que un número sea el producto de otros dos números no hubiera sido construido, no sería posible aprehender un algoritmo como algo que calcula de manera efectiva los valores de la multiplicación.

Existe, por lo tanto, una dependencia cognitiva importante de la computabilidad efectiva con respecto a los conceptos computables. ¿Podemos pensar que todo concepto computable se relaciona de este modo con algún algoritmo? Es decir, ¿existe un algoritmo para todo concepto computable de tal forma que este concepto proporcione la base cognitiva del algoritmo como un programa para calcular una función? La posibilidad de una respuesta positiva a este problema es lo que ahora quiero considerar. Exploraré la idea de que, una vez que se intuye el contenido de un concepto computable, es posible expresarlo en cierto algoritmo; esto es, consideraré la posibilidad de que, en principio, sea posible formular un algoritmo cuya ejecución clasificará, identificará o relacionará objetos del mismo modo en que lo haría el concepto asociado al algoritmo, y ver el conjunto de instrucciones componentes del algoritmo como una expresión lingüística del contenido del concepto. Diferentes algoritmos asociados al mismo concepto constituirán formas diferentes de expresar el mismo conte-

determinan qué conceptos pueden ser formados: el constructivo y el holístico. Para detalles sobre estas dos versiones, véase Cocchiarella 1986.

nido involucrado en el concepto. Sobran ejemplos para ilustrar lo que estoy planteando aquí. Considérese el concepto de que un número sea la suma de otros dos números: una vez intuido el contenido de este concepto, es posible, en principio, formular un algoritmo que proporcione la adición de dos números. Y ese concepto de suma puede ser expresado por un número infinito de algoritmos de Turing.[35]

Si asumimos la conexión entre algoritmos y conceptos computables expuesta anteriormente y la tesis Turing-Church, se seguirá, por una parte, que para todo problema Turing-computable con respecto a cierta entidad, se podrán formar uno o varios conceptos computables cuya ejecución permitiría el cómputo del mismo problema. Por otra parte, tendremos que concluir que si se pueden formar conceptos computables que posibiliten el cómputo de un problema dado con respecto a cierta entidad, debería haber una máquina de Turing cuya ejecución permitiera computar el mismo problema.

Dadas las consecuencias anteriores del nexo entre algoritmos y conceptos computables, podría pensarse que la idea de conceptos computables es innecesaria; es decir, se podría creer que el papel de los conceptos computables en la explicación general de la computabilidad carece de importancia, pues los algoritmos bastarían (por sí solos) para dar cuenta de ella. Mostraré que esto no es así, y para ello no hace falta más que tomar en cuenta los predicados T-computables —entidades definidas como computables por la teoría clásica—. En otros términos, señalaré entidades que no serían computables desde el enfoque conceptualista, pero sí desde la perspectiva de la computabilidad efectiva.

Los predicados (parcialmente) Turing-computables son expresiones de predicados cuyas extensiones constituyen el dominio de funciones (parcialmente) Turing-computables. Considérese la extensión de cualquier predicado que expresa un concepto computable. Por lo supuesto en los párrafos anteriores, debería haber un algoritmo (y por la tesis de Church, una máquina de Turing) que computara la función característica de tal extensión. En consecuencia, todo predicado que expresa un concepto computable será (parcialmente) T-computable. La pregunta que surge, naturalmente, es si todo predicado T-computable expresa un concepto computable.

Se puede ver, fácilmente, que muchos predicados (parcialmente) T-computables representan conceptos computables, toda vez que ob-

[35] Es importante indicar que en Pylyshyn 1986 se ofrece una interpretación realista de la relación aquí descrita entre algoritmos y conceptos computables. Los diversos programas computacionales cuyas entradas y salidas son las mismas tendrían en su contenido una entidad en común independiente del pensamiento.

servamos la construcción de muchos predicados numéricos en, por ejemplo, Kleene 1952 y Davis 1982a. Es más, para todo número natural, podemos encontrar predicados monádicos T-computables que representan conceptos computables. Sin embargo, dado que la noción de predicado-T tiene un componente extensional, es lógicamente posible (de acuerdo con los supuestos anteriores) que un predicado represente un concepto no computable y sea T-computable a la vez (en la medida en que su extensión sea el dominio de una función T-computable). Así, es lógicamente posible poseer predicados T-computables que representan conceptos que, al no ser computables, no nos permiten por sí mismos aplicar de forma efectiva los predicados correspondientes.

Aplicar de forma efectiva una de la expresiones a las que aludimos anteriormente requeriría conocer un algoritmo asociado a la función cuyo dominio es la extensión del predicado (o poseer el concepto computable conectado con tal algoritmo) y saber que el dominio de la función es la extensión del predicado (o que los objetos que caen bajo el concepto que representa el predicado son los mismos objetos que caen bajo el concepto computable asociado al algoritmo). Pero, de hecho, existen predicados T de los cuales ignoramos todo esto. A manera de ejemplo, considérese el predicado [λx (x es un número natural y la conjetura de Goldbach es verdadera)].[36] Su extensión es el conjunto vacío o el conjunto de los números naturales y, de este modo, se considera un predicado T, ya que su extensión sería el dominio de una función T-computable: si es vacía, sería el dominio de la función constante $g(x) = 0$; si no es vacía, sería el dominio de la función constante $f(x) = 1$. En cualquiera de los dos casos, la función correspondiente es recursiva primitiva y, por ende, el predicado en cuestión es un predicado T. Sin embargo, el concepto que ese predicado representa es claramente no computable y, por ello, tal predicado no sería computable para el conceptualismo.[37] Y esto

[36] Uso aquí el operador lambda para expresar también funciones proposicionales. Este predicado puede ser leído como "ser un número natural tal que la conjetura de Goldbach es verdadera".

[37] Por los supuestos anteriores, todo predicado T se encuentra claramente asociado a conceptos computables sólo por el hecho de estar ligado a máquinas de Turing: los predicados T se conectan con aquellos conceptos computables que corresponden a máquinas de Turing que computan las funciones cuyos dominios son las extensiones de los predicados T. Por ende, para todo predicado T habría un concepto computable (a saber, el concepto asociado al algoritmo de Turing) bajo el cual caen aquellos objetos de la extensión del predicado. En consecuencia, si un predicado T representara un concepto no computable, todavía habría un concepto extensionalmente equivalente al concepto no computable. Sin embargo, el problema subsiste: tenemos

muestra que la computabilidad desde el punto de vista clásico presenta como computables entidades que no son tales desde la perspectiva conceptualista. No podemos librarnos de los conceptos computables asumiendo que su contenido se puede expresar en algoritmos.

Las nociones de concepto totalmente computable y concepto semicomputable deben formar parte de nuestra caracterización general de las capacidades cognitivas de los seres humanos y de las bases cognitivas de los algoritmos. Sin los conceptos computables, los algoritmos no podrían ser vistos como instrucciones efectivas de funciones, sino como conjuntos de instrucciones para aparear un ingreso de datos con una salida de datos. He asumido que es posible expresar el contenido de los conceptos por medio de algoritmos y, de acuerdo con esta posibilidad, ha de existir una relación interna entre los algoritmos y los conceptos: los algoritmos expresan el contenido de los conceptos y los conceptos forman la base cognitiva de los algoritmos.

El supuesto de que los contenidos de los conceptos computables puedan ser expresados por algoritmos es, obviamente, un supuesto empírico que tendrá que esperar su momento para ser refutado o confirmado. Ahora, para finalizar, quiero contemplar la posibilidad de que hubiera conceptos computables cuyos contenidos no pudieran ser capturados por algoritmos. Esa posibilidad no implica que algunos aspectos de esos mismos contenidos no puedan ser expresados en un lenguaje; pero si tal expresión fuera posible, ésta no tendría las características de un algoritmo. El lenguaje serviría, en este caso, como medio para guiarnos hacia un concepto computable, sin que esto implicara la presentación de un método efectivo. Por otra parte, si asumiéramos que hay funciones en extensión correlacionadas con los conceptos computables (como sus correlatos), tendríamos que considerar la posibilidad de que existiesen funciones computables (en extensión) que no pudieran estar contempladas por la teoría clásica de la computabilidad. Ahora bien, esto no necesariamente significaría un rechazo de la tesis de Church, pues, como vimos, la tesis se refiere a las funciones computables en forma efectiva, esto es, aquellas cuya computabilidad requiere la existencia de un algoritmo.

La idea de un concepto computable no algorítmico (esto es, de un concepto para el cual no existiría un algoritmo que expresara su contenido) sugiere, obviamente, la idea de un agente capaz de ofrecer, en principio, una respuesta, en un tiempo finito sin apelar

predicados computables en forma efectiva o clásica, pero que no son computables desde el punto de vista conceptualista ya que el hablante no puede aplicarlos en forma efectiva (al no representar un concepto computable).

a elementos de azar o ingenio, a un problema determinado mediante el ejercicio de tal concepto sin que se pueda, en principio, hacer corresponder tal proceso con la implementación de un algoritmo. Aquí el agente computacional ejercería su concepto y la evidencia mostraría que estaríamos lidiando con un proceso computacional, pero no habría posibilidad de un cómputo clásico (esto es, de un cómputo mediante un algoritmo) que ofreciera la misma respuesta al problema. En este trabajo no voy a ahondar en cuál debe ser la naturaleza de esa evidencia; sólo asumiré que tal evidencia es posible. Pero, dado este supuesto, sería importante considerar (de manera hipotética) en qué condiciones se podría presentar la situación descrita anteriormente, con el fin de guiar la investigación empírica posible.

Ahora bien, podemos dar una respuesta, de forma inmediata y obvia, al problema de cuándo un proceso no puede ser representado por un proceso de implementación de un algoritmo, negando uno que otro aspecto de la computabilidad efectiva. De lo que se trata, más bien, es de presentar condiciones coherentes con el contexto filosófico del proceso en cuestión y relevantes para el mismo; y, en nuestro caso, para el conceptualismo realista. Siguiendo esta directriz, expondré brevemente, y a manera de ejemplo, dos posibles condiciones (hipotéticas) en las cuales no podría existir un proceso computacional correspondiente a un proceso del ejercicio de un concepto computable. Tales condiciones se centrarán en los pasos de un proceso de cómputo efectivo, o bien en las operaciones que guían el proceso mismo, y mi exposición será más bien esquemática; dejaré el análisis más profundo para otro trabajo.

La idea de un agente capaz de efectuar un cómputo sin que esto implique la posibilidad de atomizar los pasos del cómputo es completamente coherente con la noción conceptualista de computabilidad; pero además es sugerida por la idea misma expresada al final de la sección 3. Ahí manifesté que el ejercicio de un concepto computable no requiere el seguimiento de instrucciones y que podría verse como un proceso continuo que tal vez no podría ser representado por una secuencia finita de pasos. Una situación en la que se puede concebir esto último es aquella en la cual primero le asignamos cierto segmento de tiempo al ejercicio del concepto computable. Luego, cada vez que tomemos una parte del segmento y la conceptualicemos como un paso en el ejercicio del concepto, tendremos que subdividirla, a su vez, en otros dos segmentos que corresponderían a un paso intermedio entre los dos puntos, el cual se conceptualizaría como un paso

en el ejercicio del concepto; en cada uno de los dos nuevos segmentos será necesario encontrar un paso adicional conceptualizado como un paso en el ejercicio del concepto, y así sucesivamente hasta el infinito. Habría que explicar qué procesos conceptuales nos llevarían a conceptualizar cada segmento de tal forma que nos dirigiera a una conceptualización posterior de otros dos segmentos para que esta situación fuera plausible desde el punto de vista conceptualista.

Ahora bien, cabe la posibilidad de que cada uno del número infinito de subprocesos en el ejercicio del concepto computable fuera capturado por operaciones atómicas. Esto es, para cada paso de la infinita cantidad posible de partes componentes del proceso del ejercicio de un concepto, habría una operación atómica que lo justificara. No sería necesario asumir que tales operaciones se tomarían de un conjunto infinito. Perfectamente, la totalidad de ellas podrían constituirse en un conjunto finito; pero, obviamente, el cómputo mismo no constituiría una secuencia finita resultante de operaciones atómicas. Y esto es imprescindible para que el proceso pueda considerarse un proceso de implementación de un algoritmo.

En la situación anterior, he enfocado el proceso mismo para ejercer un concepto computable: el ejercicio de un concepto computable se da en un tiempo finito, pero el proceso mismo no puede ser analizado sin caer en un número infinito de subprocesos. Ahora bien, ¿que pasaría si asumiéramos que es posible una división en forma discreta del ejercicio del concepto computable? Esto nos dirigiría a la posibilidad de no contar con operaciones atómicas capaces de justificar los diferentes momentos en el ejercicio del concepto computable. Es decir, aunque veamos el ejercicio de un concepto en forma discreta, la imposibilidad de representarlo mediante la implementación de un algoritmo podría deberse a la no existencia de un conjunto finito de operaciones atómicas que nos permitiera contemplar la división discreta como producto de la aplicación de algunas de las operaciones de ese conjunto. Estaríamos ante la posibilidad de un agente capaz de efectuar acciones computacionales, cuyo proceso no podría ser atomizado en subprocesos mecánicos susceptibles de ser capturados por operaciones de un conjunto finito de operaciones atómicas. Cada uno de los subprocesos de ese conjunto finito podría ser conceptualizado, pero no por un concepto o conceptos cuyo contenido correspondan a una operación atómica como las contempladas en las máquinas de Turing. Uno de los problemas importantes para investigar en esta situación es cuál sería la naturaleza de los conceptos que pueden capturar los subprocesos y que no pueden ser construidos a partir de conceptos correspondientes a operaciones atómicas.

6. *Consideraciones finales*

He caracterizado un enfoque conceptualista de la noción de computabilidad y lo he contrastado con la interpretación clásica conocida como *computabilidad efectiva*. Como se mostró, esta última interpretación descansa en la noción de algoritmo, a diferencia de la primera en la que el concepto central es el de concepto computable. Es por ello que la interpretación clásica exige la existencia de un lenguaje cuyo uso, en principio, pueda ser utilizado por un ser humano, pues se supone que una idealización de éste constituye el agente ideal que ha de implementar el algoritmo.

He examinado las relaciones entre la interpretación conceptualista y la clásica al explorar posibles nexos entre algoritmos y conceptos computables. En una primera consideración, supuse que los algoritmos lograban expresar el contenido de los conceptos computables y que éstos, a su vez, constituían la base cognitiva a partir de la cual es posible conceptualizar a los primeros como si calcularan funciones computables. A partir de estos supuestos mostré que los conceptos computables no son entidades innecesarias y por lo tanto tendrían que ser tomados en cuenta a la hora de formular una teoría de la computabilidad que involucrara algoritmos. Finalmente exploré la idea de que los contenidos de algunos conceptos computables no pudieran ser capturados por algoritmos y contemplé condiciones conceptualmente consistentes en las cuales el ejercicio de un concepto computable no puede ser representado por un proceso de implementación de algoritmo.

Cualesquiera que sean las relaciones de los conceptos computables con los algoritmos, creo que en este artículo se han proporcionado bases filosóficas para el desarrollo formal de una teoría de la computabilidad alternativa a la clásica. En esta teoría podrían retomarse muchos de los conceptos de la interpretación clásica y verlos a la luz de la fundamentación filosófica que planteo. Por ejemplo, los predicados T podrían ser redefinidos como aquellos predicados que representan conceptos computables y cuyas extensiones son el dominio de funciones Turing-computables.[38] Sobre la base de esta definición, habría que explorar cuáles de los principios sobre tales predicados de la teoría clásica se sostendrían y si acaso existen algunos nuevos e

[38] Esta definición no supondría que a todo concepto computable le corresponde una máquina de Turing, la cual expresaría su contenido. De lo contrario, estaría de más la condición (en la definición descrita) de que a todo predicado T le corresponde una máquina de Turing que calcularía su extensión.

interesantes por formular. Tendríamos, en este caso, una teoría de la computabilidad intensional conceptualista.

Apéndice: la teoría presentacionista y la teoría intuicionista de la computabilidad

La teoría conceptualista de la computabilidad aquí propuesta parecería tener aspectos análogos con dos teorías no clásicas de la computabilidad: la teoría intuicionista y la (que aquí llamaremos) *teoría presentacionista* de la computabilidad. Por ello, para finalizar, quiero esbozar algunas de las diferencias que guardan estas teorías con la conceptualista.

(i) La teoría presentacionista expuesta en Shapiro 1980 y en Nelson 1987 se basa en la distinción entre funciones (en extensión) y descripciones de éstas en un lenguaje interpretado, las cuales son llamadas *presentaciones* de funciones.[39] En otros términos, las funciones son conjuntos de pares ordenados y las presentaciones expresiones lingüísticas que describen las funciones.[40] Por otra parte, se distingue efectividad de computabilidad en cuanto atributos. La noción de efectividad es un atributo de las presentaciones y no de las funciones: una presentación de una función f es *efectiva* si y sólo si sugiere al agente computacional un algoritmo para determinar $f(n)$ a partir de n. A pesar de la importancia del concepto de que una presentación sugiera a un agente computacional un algoritmo, en la teoría presentacionista no hay una explicación somera de ese concepto.

Computabilidad es un atributo de funciones: una función es *computable* si y sólo si existe un algoritmo para determinar $f(n)$ a partir de n. Finalmente, una función que es computable en forma efectiva es una función computable para la cual existe una presentación efectiva. La tesis de Turing-Church se enuncia del siguiente modo: *cualquier función computable f que tiene una presentación efectiva es recursiva*.

Dejando de lado el hecho de que las funciones en nuestra teoría no son entidades abstractas y que en la teoría presentacionista sí lo son, esta teoría difiere evidentemente de la conceptualista en su dependencia del lenguaje. De acuerdo con la perspectiva presentacionista, el que una función sea computable en forma efectiva no

[39] Véase Feferman 1985 para una discusión de matemáticas intensionales donde el concepto de presentación de una entidad matemática es central.

[40] Nuestro ejemplo de la página 29 sería una presentación no efectiva de una función computable. La expresión $[\lambda x(x+1)]$ es una presentación efectiva de la función computable $\{(x, x+1) \mid x \in \omega\}$.

depende de nuestras capacidades cognitivas, sino, más bien, de las posibles combinaciones de los términos de un lenguaje, combinaciones que no tienen que haber sido construidas por un posible agente. Un algoritmo es un conjunto de instrucciones en un lenguaje y una presentación es una expresión que describe una función. En ambos casos, son combinaciones posibles del alfabeto de un lenguaje.

(ii) La teoría intuicionista forma parte de toda una familia de teorías de corte constructivista, las cuales requieren que cualquier prueba de existencia de entidades matemáticas sea constructiva, esto es, que se construya el objeto cuya existencia se afirma o se proporcione un método (con características finitistas) para lograr tal construcción.[41] En el contexto de la computabilidad, lo que se exige es que cualquier afirmación de existencia de un algoritmo se justifique constructivamente. Y esto marca una diferencia de la teoría conceptualista con la intuicionista: en la primera no se exige que la prueba de existencia de un concepto computable sea constructiva. En la teoría conceptualista que aquí he formulado, no se han planteado criterios de existencia intuicionistas: es cierto que los conceptos computables sólo existen como capacidades cognitivas en los seres humanos pero, en principio, pueden ser formados y la prueba de esto no tiene que ser constructiva. No se exigen métodos finitistas de prueba de la existencia actual o en principio de un concepto computable.

Tómese, por ejemplo, el caso de los predicados T. En la teoría conceptualista se exige que cualquiera de tales predicados represente un concepto computable (lo cual nos permitiría aplicar de forma efectiva el predicado) y esto requiere, a la vez, la formación de capacidades cognitivas. No obstante, esta exigencia no significa que la prueba de existencia de tales capacidades tenga que ser, en principio, constructiva. En otros términos, queda abierta la posibilidad de uso de métodos de prueba no intuicionistas para mostrar que, en principio, el concepto computable puede ser o ha sido formado.[42]

BIBLIOGRAFÍA

Bowie, G.L., 1973, "An Argument Against Church's Thesis", *The Journal of Philosophy*, vol. LXX, no. 8, pp. 66–76.

[41] Para una discusión del concepto de computabilidad en el contexto intuicionista, véanse Péter 1959 y McCarty 1987.

[42] Agradezco a Manuel Arce (Universidad de Costa Rica), Axel Barceló, Raymundo Morado, Maite Ezcurdia (UNAM) y a los árbitros anónimos de *Crítica* sus comentarios y sugerencias a una versión previa de este artículo.

Church, A., 1941, *The Calculus of Lambda Conversion*, Princeton University Press, Nueva Jersey.

———, 1937, "Review of Turing 1936", *Journal of Symbolic Logic*, vol. 2, no. 1, pp. 42–43.

———, 1936, "An Unsolvable Problem of Elementary Number Theory", *American Journal of Mathematics*, vol. 58, pp. 345–363.

———, 1935, "An Unsolvable Problem of Elementary Number Theory" (Abstract), *Bulletin of the American Mathematical Society*, vol. 41, pp. 333.

Cocchiarella, N., 1986, *Logical Investigations of Predication Theory and the Problem of Univesals*, Bibliopolis, Nápoles.

Cutland, N., 1980, *Computability: An Introduction to Recursive Function Theory*, Cambridge University Press, Cambridge.

Davis, M., 1982a, *Computability and Unsolvability*, Dover Publications, Nueva York.

———, 1982b, "Why Gödel Didn't Have Church's Thesis", *Information and Control*, vol. 54, pp. 3–24.

———, 1965, *The Undecidable*, Raven Press, Nueva York.

Enderton, H., 1977, "Elements of Recursion Theory", en John Barwise (comp.), *Handbook of Mathematical Logic*, North Holland, Amsterdam.

Epstein R. y W. Carnielli, 2000, *Computability*,Wadsworth, Pacific Grove.

Feferman, S., 1985, "Intensionality in Mathematics", *Journal of Philosophical Logic*, vol. 14, no. 1, pp. 41–55.

Freund, M., 2004, "A Modal Sortal Logic", *Journal of Philosophical Logic*, vol. 33, no. 3, pp. 237–270.

———, 2002, "Conceptual Realism and Interpretation", *Protosociology. An International Journal for Interdisciplinary Research*, vol. 17, no. 1, pp. 119–137.

———, 2001, "A Temporal Logic for Sortals", *Studia Logica*, vol. 69, no. 3, pp. 351–380.

———, 2000, "A Complete and Consistent Formal System for Sortals", *Studia Logica*, vol. 65, no. 3, pp. 367–381.

———, 1996, "A Minimal Logical System for Computable Concepts and Effective Knowability", *Logique et Analyse*, vol. 34, no. 4, pp. 339–366.

Galton, A., 1996, "The Church-Turing Thesis: its Nature and Status", en A. Clark y P. Millican (comps.), *Machines and Thought*, Oxford University Press, Oxford.

Gandy, R., 1988, "The Confluence of Ideas in 1936", en R. Herken (comp.), *The Universal Turing Machine: A Half-Century Survey*, Oxford University Press, Oxford.

———, 1980, "Church's Thesis and Principles for Mechanisms", en *The Kleene Symposium*, North Holland, Amsterdam, pp. 123–148.

Gödel, K., 1995, *Collected Works. III: Unpublished Essays and Lectures*, ed. S. Feferman *et al.*, Oxford University Press, Oxford.

Gödel, K., 1946, "Remarks before the Princeton Bicentennial Conference on Problems in Mathematics 1946", en Davis 1965, pp. 84–88.
———, 1934, "On Undecidable Propositions of Formal Mathematical Systems", en Davis 1965, pp. 41–74.
Hermes, H., 1969, *Enumerability, Decidability, Computability*, Springer, Berlín.
Kálmar, L., 1959, "An Argument against the Plausibility of Church's Thesis", en A. Heyting (comp.), *Constructivity in Mathematics*, North Holland, Amsterdam, pp. 72–80.
Kleene, S.C., 1981, "Origins of Recursive Function Theory", *Annals of Historical Computing*, vol. 3, no. 1, pp. 52–66.
———, 1967, *Mathematical Logic*, John Wiley and Sons, Nueva York.
———, 1952, *Introduction to Metamathematics*, North Holland, Groningen.
———, 1935a, "General Recursive Functions of Natural Numbers", *Bulletin of the American Mathematical Society*, vol. 41, p. 489.
———, 1935b, "λ-Definability and Recursiveness", *Bulletin of the American Mathematical Society*, vol. 41, p. 489.
———, 1935c, "General Recursive Functions of Natural Numbers", *Mathematische Annalen*, vol. 112, pp. 727–742.
———, 1935d, "λ-Definability and Recursiveness", *Duke Mathematical Journal*, vol. 2, pp. 340–353.
McCarty, C., 1987, "Variations on a Thesis: Intuitionism and Computability", *Notre Dame Journal of Formal Logic*, vol. 28, no. 4, pp. 536–580.
Mendelson, E., 1990, "Second Thoughts About Church's Thesis and Mathematical Proofs", *The Journal of Philosophy*, vol. 87, no. 5, pp. 225–233.
———, 1963, "On Some Recent Criticism of Church's Thesis", *Notre Dame Journal of Formal Logic*, vol. 4, no. 3, pp. 201–205.
Nelson, R.J., 1987, "Church's Thesis and Cognitive Science", *Notre Dame Journal of Formal Logic*, vol. 28, no. 4, pp. 581–614.
Péter, R., 1959, "Rekursivität und Konstruktivität", en A. Heyting (comp.), *Constructivity in Mathematics*, North Holland, Amsterdam, pp. 226–233.
Post, Emil, 1936, "Finite Combinatory Processes. Formulation I.", *Journal of Symbolic Logic*, vol. 1, pp. 103–105.
Pylyshyn, Z.W., 1986, *Computation and Cognition*, The MIT Press, Cambridge, Mass.
Rogers, H., 1967, *Theory of Recursive Functions and Effective Computability*, McGraw-Hill, Nueva York.
Ross, D., 1974, "Church's Thesis: What its Difficulties Are and Are Not", *Journal of Philosophy*, vol. 71, no. 15, pp. 515–525.
Shapiro, S., 1981, "Understanding Church's Thesis", *Journal of Philosophical Logic*, vol. 10, no. 3, pp. 353–365.
———, 1980, "On the Notion of Effectiveness", *History and Philosophy of Logic*, vol. 1, pp. 209–230.

Sieg, W., 1997, "Step by Recursive Step: Church's Analysis of Effective Calculability", *The Bulletin of Symbolic Logic*, vol. 3, no. 2, pp. 154–180.

———, 1994, "Mechanical Procedures and Mathematical Experience", en A. George (comp.), *Mathematics and Mind*, Oxford University Press, Oxford.

Sieg, W. y J. Byrnes, 1995, "K-Graph Machines: Generalizing Turing's Machines and Arguments", en P. Hájek (comp.), *Lecture Notes in Logic*, vol. 6, Springer, Berlín.

Soare, R.I., 1996, "Computability and Recursion", *The Bulletin of Symbolic Logic*, vol. 2, no. 3, pp. 284–321.

Turing, A.M., 1939, "Systems of Logic Based on Ordinals", *Proceedings of the London Mathematical Society*, vol. 45, no. 52, pp. 161–228.

———, 1936, "On Computable Numbers, with an Application to the Entscheidungs-Problem", *Proceedings of the London Mathematical Society*, serie 2, vol. 42, pp. 230–265.

Recibido el 12 de agosto de 2004; revisado el 12 de enero de 2006; aceptado el 1 de febrero de 2006.