XVI CONGRESO NACIONAL DE FILOSOFÍA (AFRA)

GRISELDA PARERA
DIANA MARÍA LÓPEZ
SOL YUAN
(COMPILADORES)

UNIVERSIDAD NACIONAL DEL LITORAL
FACULTAD DE HUMANIDADES Y CIENCIAS

Sobre computaciones y reglas

JAVIER BLANCO / PÍO GARCÍA
Universidad Nacional de Córdoba

Introducción

Desde posiciones muy diferentes se ha puesto en cuestión tanto el concepto de computación como el alcance de la tesis Church-Turing (esto es la tesis que afirma la adecuación de la caracterización de computación a partir de la equivalencia extensional de los formalismos utilizados). Entre dichas posiciones se puede señalar el proyecto de comprender a la mente como una computadora (Piccinini 2009), la propuesta de que cualquier sistema de la suficiente complejidad realiza cualquier computación (pancomputacionalismo) e incluso la extensión del argumento wittgensteniano de "seguir una regla" al ámbito de la computación (Shanker 1987).

Estas controversias parecen apuntar a un *mismo* problema: qué se entiende por computación. Y efectivamente esta parece ser la discusión central. Pero, por la forma en la cual se han desarrollado estas discusiones, se hace referencia aquí a problemas *diferentes*. Por un lado está la cuestión central de qué es computar (Copeland 1996), en segundo lugar el problema de qué es una computadora, en tercer lugar cuáles son los criterios que me permiten identificar cuándo un sistema computa, finalmente está el problema de la relación entre los aspectos físicos y abstractos de la computación (Putnam 1988). Marcar la relativa independencia entre estos problemas significa que la respuesta que se dé a alguno de estos no necesariamente resuelve las otras cuestiones (aunque, insistimos, estén íntimamente relacionados).

En este trabajo avanzaremos sobre una estrategia posible para abordar aspectos de los dos primeros problemas (qué es computar y qué es una computadora). Consideraremos el primer problema en términos del segundo. Esto es: tomaremos el concepto de computación a partir de una caracterización de la máquina "computadora". Siguiendo la sugerencia de Canguilhem (1976) de que los diferentes tipos de mecanicismos pueden ser entendidos a partir de la "máquina" que se toma como modelo, proponemos que esta "máquina modelo" estaría representada no sólo por la llamada "máquina de Turing" sino por la "máquina universal". Cuando se incluye esta última en la discusión aparece, en principio, un horizonte de posibilidades di-

ferente en donde se pueden analizar algunos de los problemas anteriores. Si bien la máquina universal es una máquina de Turing particular, cuando se consideran sus características específicas pueden plantearse mejor varias cuestiones que quedan implícitas en cierta engañosa simpleza de estas máquinas. Puesto en otros términos, nuestra intención es responder a la pregunta ¿qué aspectos centrales de la noción de computación aparecen explicitados cuando se toma en consideración a la máquina universal?.

Pancomputacionalismo y otros desafíos

¿De qué forma se ha problematizado el concepto de computación?. En el ámbito de la filosofía de la mente se ha cuestionado la tesis "computacionalista" en términos de su capacidad explicativa. Una forma de hacer esto es mostrando que la explicación en términos de mecanismos computacionales no agrega nada relevante a nuestra comprensión de la mente. Si cualquier sistema de la suficiente complejidad puede ser descripto como una computadora, entonces no hay nada específico en el señalamiento de que la mente es una computadora (Searle 1990). Para algunos esta tesis implicaba poner en cuestión la adecuación de la caracterización de computación por parte de Turing.

La tesis del Pan-computacionalismo tuvo su origen en el trabajo de Putnam y Searle, y afirma que cualquier sistema físico con suficientes estados puede implementar cualquier computación.

"Las definiciones originales dadas por Alan Turing conforman la definición estándar de computación.. En la definición estándar de computación es difícil ver cómo evitar lo siguiente: 1. para cualquier objeto, hay una descripción de ese objeto tal que bajo dicha descripción el objeto es una computadora digital. 2. Para cualquier programa y para cualquier objeto suficientemente complejo, hay alguna descripción del objeto bajo la cual implementa el programa. Así por ejemplo la pared a mis espaldas está ahora implementado el programa Wordstar porque hay algún patrón del movimiento de las moléculas que es isomórfico con la estructura formal de Wordstar. Pero si la pared está implementando el programa Wordstar, entonces si hay una pared lo suficientemente grande, ésta puede implementar cualquier programa, incluido cualquier programa implementado en el cerebro." (Searle 1990)

Luego de este intento de trivializar la explicación computacionalista, Searle sugiere algunas soluciones:

"Creo que es posible bloquear el resultado de la realizabilidad múltiple restringiendo nuestra definición de computación. Una definición más realista de computación enfatizará aspectos tales como las relaciones causales entre los estados del programa, la programabilidad y la controlabilidad del mecanismo y su situacionalidad en el mundo real" (Searle 1990)

De hecho la mayoría de estos caminos posibles de solución han sido recorridos por los filósofos: el camino causal -en términos contrafácticos- ha sido sugerido por Copeland (1996) y por Chalmers (1996), Piccinini (2009), por otro lado ha señalado a la controlabilidad del sistema como un aspecto clave de las computadoras que sean "usables". En otros trabajos hemos argumentado en favor de la programabilidad como una característica saliente de la computación (Blanco, Garcia, Cherini 2011). La modificabilidad y plasticidad en los sistemas computacionales podrían entenderse desde la propiedad más pertinente de la programabilidad.

En términos más generales, para algunos investigadores, solucionar este problema significaba encontrar algún conjunto de características o propiedades específicos de la computación que excluyan la aplicación indiscriminada de la tesis pan-computacionalista. Para otros autores, la pregunta era cómo especificar de manera adecuada la realización física de un sistema computacional. Así, por ejemplo, Piccinini (2007) presenta el concepto de mecanismo como una propiedad específica de la computación. Copland (1996) sugiere el concepto de arquitectura (junto con la idea de modelos honestos) como una respuesta. Scheutz (1999) trabaja la idea de "realización de una función" y sus condiciones físicas como concepto clave contra la pan-computacionalismo. Chalmers (1996) trata de evitar las consecuencias de esta tesis diciendo que el argumento de Putnam carece de una teoría adecuada de la implementación.

Desde el campo de la filosofía del lenguaje se ha sugerido que en la propuesta original de Turing hay una "confusión" entre una tesis matemática y otra filosófica (Shanker 1987); y que a partir de una posición wittgensteniana se podrían marcar los límites de la idea de computación en términos de seguir una regla. El articulo de Turing de 1936 sería un trabajo "híbrido", en donde aparece tanto la noción de cálculo efectivo como la anticipación de la idea de las máquinas que piensan de 1950. De acuerdo con la interpretación de Shanker, seguir una regla involucra aspectos normativos que no aparecen en la idea de máquina de Turing; relegando de esta manera la computación a un ámbito descriptivo se marcan límites - en principio- de lo que una máquina puede hacer. Habría de esta manera una diferencia entre las actividades de "calcular" (normativo) y de "contar" (empírico). Esta diferencia podría explicar la expresión de Wittgenstein acerca de que las máquinas de

Turing son en realidad "humanos que calculan". Más allá de si la versión de Shanker es una interpretación adecuada del pensamiento de Wittgenstein, puede ser considerada un desafío acerca de los límites de la noción de computación.

Finalmente se pueden señalar algunos de los desafíos que se han presentado a la noción de computación a partir del cuestionamiento de la tesis Church-Turing. Las computadoras "físicas" podrían computar más funciones que las máquinas de Turing. A su vez esta tesis de Church-Turing física podría tomar una forma general (*bold*) o podría restringirse a aquellas computadoras que sean manipulables de una forma adecuada (*modest*) (Cfr. Piccinini 2011)

Si bien son importantes los desafíos presentados por las versiones llamadas "físicas" de la tesis Church-Turing o por la hipercomputación, en tanto son de una naturaleza diferente a los problemas discutidos no los consideraremos en el presente trabajo.

Máquina de Turing y Máquina Universal

Hay un acuerdo generalizado de que la noción de computación ha sido caracterizada de manera sustantiva por Turing en su artículo acerca de los números computables de 1936 (Turing 1936). El significado de lo que implica calcular de manera efectiva se entendería a partir de ahora como la ejecución paso a paso del funcionamiento de máquinas abstractas. Este acuerdo parece ir más allá de encontrar un formalismo para caracterizar una noción. Así parece entenderlo Godel cuando dice que con Turing encontramos por primera vez una noción "absoluta" de computación.[1]

Mucho se ha dicho sobre el alcance de la noción de "máquina" - y de "máquina de Turing"- para entender a la de "computación". Menos explorado es el eventual aporte de la idea de máquina universal (MU) que Turing presenta en el mismo artículo de 1936. Por supuesto que ha sido ampliamente reconocida la importancia de las MU como una nota distintiva de la propuesta de Turing (Davis 2000). Lo que queremos problematizar es el aporte de las MU *a la noción misma de computación*. Este aporte parece ir más allá de una mera sofisticación de las máquinas de Turing o de una extensión de su campo de aplicación. Las MU muestran algunos aspectos

[1] 'Remarks Before the Princeton Bicentennial Conference' p. 84. Godel habla de una definición absoluta de una noción epistémica interesante. Es un texto de 1946, pero publicado recién en una compilación de Davis en 1965.

interesantes de lo que las MT *pueden* hacer. Y estos aspectos están vinculados con una caracterización de la computación.

Una máquina de Turing está definida a partir de un conjunto finito de estados, un alfabeto finito de símbolos de entrada (se suele usar un alfabeto binario) y una función de transición, la cual suele ser presentada como una tabla de transición. Los datos de entrada vienen en una cinta infinita (pero con una cantidad finita de símbolos no nulos), sobre la cual habrá una posición diferenciada que indicará el símbolo corriente. Cada una de las filas de la tabla indica para cada estado y símbolo de entrada dados cómo se cambia el símbolo corriente de la cinta, si se mueve para la izquierda o la derecha la posición corriente, y cuál será el nuevo estado.

El "computador" presentado por Turing para realizar las computaciones prescriptas por sus máquinas abstractas es una persona equipada con lápiz y papel que toma una tabla de transición como codificación de un comportamiento cuyos datos de entrada están la cinta, y va aplicando mecánicamente los pasos prescriptos en esa tabla. Como decíamos, cada paso indica posibles modificaciones a una posición de la cinta, un posible cambio de posición y cuál es el siguiente estado. Si se llega a un estado y un símbolo de entrada para el cual no hay ninguna regla en la tabla, el programa termina.

La MU es una máquina particular (dada por su propia tabla de transición) cuyo comportamiento consiste inicialmente en recorrer la cinta leyendo el código de una máquina dada, y luego comportándose como dicha máquina tomando como entrada el resto de la cinta. Esta posibilidad depende de que se pueda codificar una MT y que luego esta codificación sea parte de los “datos” de entrada de otra MT particular que ahora funciona como una MU. En la MU pueden plantearse con mayor claridad algunas propiedades que consideramos relevantes de la computación como la programabilidad.

Se puede hacer una distinción entre sistemas más interesantes que otros, donde “interesante” está vinculado a la capacidad de cómputo y a la programabilidad de los sistemas. Ambas nociones admiten grados. Es más, se puede construir una jerarquía de sistemas computacionales (Blanco, Cherini, Diller, Garcia 2011) tomando en consideración su comportamiento o los lenguajes de programación que admiten. Podemos caracterizar a la programabilidad como la capacidad de comportarse de formas sustancialmente distintas de acuerdo con la entrada del sistema.

Los intérpretes y compiladores en computación surgieron porque se necesitaba algún tipo de traducción entre niveles (en nuestro caso se pude decir entre “máquinas de Turing”, pero puede ser, por ejemplo, entre un lenguaje de alto nivel y el lenguaje de máquina). Por esta razón Davis dice que la MU es un primer ejemplo de

intérprete, porque su función es tomar como entrada la descripción de una máquina de Turing y los datos de esa máquina, para luego comportarse como dicha máquina. En este sentido se puede decir que un intérprete realiza una tarea de traducción entre distintos niveles. Y si la programabilidad, como decíamos arriba, se puede caracterizar como la capacidad de comportarse de maneras sustancialmente distintas de acuerdo con una entrada, la MU es un ejemplo privilegiado de sistema computacional programable.[2]

La distinción entre hardware, programa y datos deja de ser nítida, en algunos casos se vuelve una noción relacional[3]. Es importante subrayar que aquí tomamos algunos de los sentidos de la relación hardware-software. Uno de esos sentidos se refiere a la modificabilidad: aquello que regula el comportamiento y que es más directamente cambiable es el software. Otro sentido consiste en una idea muy similar pero con connotaciones "ontológicas": el hardware es aquello que constituye el soporte efectivo de las instrucciones y los datos (el "input" en general) y que permite la generación del comportamiento respectivo. La codificación de una MU, permite, a una máquina particular, cumplir ambas funciones.

La MU es ella misma una máquina de Turing, por lo cual su comportamiento está determinado por la tabla de transición. Sin embargo, podemos considerar a la MU en el punto en que ya leyó la codificación de una máquina cualquiera Z como una emulación de esta. La MU puede verse aquí como controlada o guiada por las instrucciones codificadas de Z, lo cual puede explicarse haciendo referencia a las reglas elementales que definen a MU misma, así también como al estado de la cinta de entrada de MU.

A partir de lo dicho se puede decir que una noción "mínima" de computación no parece requerir más que de las características de una máquina de Turing, pero una noción más completa ("ideal") de computación parece requerir de la noción de MU. Con la expresión "más completa" o "ideal" queremos decir que es un sistema más flexible, modificable y con propiedades distintas a una máquina de Turing. Y esta noción es ideal en tanto funciona como prescripción para la construcción de sistemas computacionales interesantes (esto es flexibles y programables). De esta forma

[2] Una respuesta apropiada al pancomputacionalismo depende de caracterizar a los sistemas computacionales como admitiendo grados.

[3] De cierta manera podemos pensar a la tabla de transición de una máquina de Turing como un programa que se aplica a los datos codificados en la cinta, la MU hace aparecer como parte de los datos al programa mismo, pudiendo pensarse a la tabla de transición de la MU como el *hardware*, ya que no es necesario cambiarla, con esa única tabla puede implementarse cualquier otra máquina sólo codificándola como parte de los datos.

se puede releer la historia de la construcción de sistemas computacionales desde fines de la década del 30 del siglo pasado.

Consideraciones finales

Una MU puede ser considerada un intérprete (en el sentido computacional). Un intérprete puede ser visto como un programa que toma otro programa y un conjunto de datos y se comporta como este último programa. En este sentido ante la pregunta de si una máquina de Turing es un intérprete, la respuesta es: depende qué aspectos consideremos. Si tomamos, como habitualmente se hace, el conjunto de reglas (tabla de transición) y los datos de entrada a partir de los cuales se genera el comportamiento, entonces una MT no es un intérprete. Sin embargo, si consideramos no sólo la tabla de transición y los datos, sino también al "computador" que manipula los datos, entonces *podemos* tener una intérprete. Decimos "podemos tener" porque depende cómo consideremos el papel del "computador". A partir de la descripción original de Turing, si el computador es un ser humano que manipula símbolos, entonces, bajo esta descripción, tenemos un intérprete. Puesto que el "computador" va a poder comportarse como cualquier máquina de Turing. Pero esta posibilidad está planteada en un nivel *permitido* por la metáfora de Turing, aunque *excluido* de su descripción explícita. En la máquina universal queda *explicitado* el papel de la máquina que ejecuta otro programa y se comporta como tal. Y esta explicitación permite la emergencia de varias propiedades interesantes a los fines de caracterizar qué es computar.

En primer lugar, el que una máquina se "ejecute" pone en cuestión la universalidad de una distinción entre hardware y software en términos ontológicos o en términos de la distinción entre algo abstracto y concreto-físico. Por supuesto, como ya hemos dicho, esta disolución de la distinción no afecta a todos los sentidos de la relación hardware-software.

En segundo lugar, en tanto en una MU se explicita una "semántica", al menos aquella que permite al programa-intérprete ejecutar *adecuadamente* al programa-máquina, hay un sentido definido en el cual se puede decir si el programa-intérprete implementa correctamente al programa-máquina. Si bien esta propiedad no da cuenta de la amplitud de la discusión presentada por Shanker a propósito de la normatividad, podría discutirse si esta noción no admitiría algún tipo de graduación. Shanker partía del supuesto de la distinción descriptivo-normativo sin más como un límite de la noción de computación propuesta por Turing. Pero uno de los

sentido de normatividad -aunque no el único por cierto- es la posibilidad de señalar cuándo un comportamiento dado es correcto. Si la MU no se comporta como la MT correspondiente (no genera los comportamientos estipulados) entonces puede que esté mal codificada.

En tercer lugar, como decíamos más arriba la propiedad de programabilidad, tan importante para discriminar entre sistemas computacionales "interesantes" y aquellos que no lo son, aparece más claramente en la máquina universal. Si la propiedad de programabilidad está vinculada con la flexibilidad de un sistema -la capacidad de comportarse de maneras sustancialmente distintas de acuerdo con un *input*-, entonces, una máquina universal muestra idealmente este aspecto de la noción de computación que una máquina de Turing (a pesar de que esta última también es programable).

En la introducción de este trabajo citábamos la sugerencia de Canguilhem acerca de la pertinencia de considerar el tipo de máquina que sirve como modelo, muchas veces implícito, para propuestas mecanicistas. Si bien en el presente trabajo no hemos analizado de manera directa la relación entre las nociones de máquina de Turing, MU y la idea más general de máquina, hemos iniciado dicho análisis poniendo el foco en la noción de MU. El mecanicismo computacional resultante a partir de este foco parece tener características distintas a la de una mera MT.

Revisar nuestras intuiciones y prejuicios a la luz del análisis de las propiedades que surgen a partir de una -cierta- ampliación de la noción de computación no es el único camino posible, pero parece fructífero.

Referencias bibliográficas

Blanco,J. Cherini, R. Garcia, P. (2011) "Convergencias y divergencias en la noción de computación" Revista Iberoamericana de Ciencia, Tecnología y Sociedad

Canguilhem, G. (1976) El conocimiento de la vida, Barcelona, Anagrama

Chalmers, D. J. (1996). "Does a Rock Implement Every Finite-State Automaton?," Synthese 108, pp. 310–333.

Copeland, B. J. (1996). What is computation? Synthese 108 (3):335-59.

Davis, M (2000)The Universal Computer, W.W. Norton & Company.

Piccinini, G (2007) "Computing Mechanisms," Philosophy of Science, 74.4, pp. 501-526

Piccinini, G. (2009). Computationalism in the Philosophy of Mind. Philosophy Compass 4 (3):515-532.

Piccinini, G. (2011) “The Physical Church-Turing Thesis: Modest or Bold?” British Journal for the Philosophy of Science, 62.4 pp. 733-769

Putnam, H. (1988). Representation and Reality. Cambridge, MA: MIT Press.

Scheutz, M. (1999),“When Physical Systems Realize Functions ”, Minds and Machines 9: 161–196.

Searle, J. (1990) 'Is the Brain a Digital Computer?', Proceedings and Addresses of the American Philosophical Association 64, 21-37.

Shanker (1987) ‘Wittgenstein versus Turing on the Nature of Church’s Thesis’, *Notre Dame Journal of Formal Logic*, vol. 28, no. 4.

Turing, A.M. (1936). "On Computable Numbers, with an Application to the Entscheidungs problem". *Proceedings of the London Mathematical Society*. 2 42: 230–65