La historia y la gramática de la recursión: Una precisión desde la obra de Wittgenstein✤

History and Grammar of Recursion: Clarifications from Wittgenstein's Work

---

Sergio Mota◈
Universidad Autónoma de Madrid- España

Recibido: 2014-05-06
Envío a pares: 2014-05-06
Aprobado por pares: 2014-06-05
Aceptado: 2014-06-07

Pensamiento y Cultura | ISSN: 0123-0999 | eISSN: 2027-5331
pensam.cult | Vol. 17-1 | Junio de 2014 | pp. 20-48
DOI: 10.5294/pecu.2014.17.1.2

✤ Quiero agradecer a los revisores anónimos de la revista *Pensamiento y Cultura* sus comentarios, los cuales han ayudado a mejorar el texto.
◈ sergio.mota.v@gmail.com

La historia y la gramática de la recursión: Una precisión desde la obra de Wittgenstein

Resumen: El objetivo principal de este trabajo es presentar el concepto de recursión, su historia y evolución dentro de la Teoría de la Computabilidad y la Lógica Matemática. En este análisis destacaré la sorprendente ausencia de Wittgenstein en los tratamientos sobre dicha noción, argumentando que no está justificada tal ausencia. De hecho mostraré que la recursión es una propiedad central en la obra de Wittgenstein, desde el Tractatus hasta su énfasis en la noción de regla gramatical. Por último, destacaré una serie de aplicaciones derivadas de la precisión y tratamiento que Wittgenstein hizo de tal noción. Dichas aplicaciones serán restringidas al ámbito de la Ciencia Cognitiva del Lenguaje.

Palabras clave: Definición por recursión, auto-inclusión, Teoría de la Computabilidad, regla gramatical, Tractatus, Wittgenstein.

History and Grammar of Recursion: Clarifications from Wittgenstein's Work

Abstract: The main goal of this paper is to introduce the concept of recursion, its history, and its evolution within Computability Theory and Mathematical Logic. In this analysis I will stress the remarkable neglect of Wittgenstein's contribution to the development of this notion, and will argue that such neglect is totally unjustified. In fact, I will show that recursion is a central property in Wittgenstein's work, from the Tractatus to his emphasis on the notion of grammatical rule. Finally, I will focus on a number of applications stemming from Wittgenstein's thoughtful reflections on the concept of recursion. Such applications will be restricted to the domain of the Cognitive Science of Language.

Key words: Recursive definition, self-embedding, Computability Theory, grammatical rule, Tractatus, Wittgenstein.

1. Historia y evolución del concepto de recursión

No hay ninguna duda de que el concepto de recursión surgió en el seno de la Teoría de la Computabilidad (una rama especializada de la Lógica Matemática). Su origen se remonta al siglo XIX y fue ya utilizado por autores como Dedekind o Peano (Soare 1996, 1999, 2009).

Sin embargo, parece que no siempre ha sido empleado de la misma manera. Así, dentro de tales disciplinas significó tanto *definición por recursión* –o definición recursiva– como *computabilidad* (y lo mismo vale para 'recursivo' y 'computable'; véase, por ejemplo, Soare 2009). Tal como nos indica Kleene (1952), una definición recursiva es un método que sirve para definir una función (o predicado).[1] Así, una función es recursiva, esto es, está definida, en un sentido técnico, por recursión cuando para definir un argumento *y* hacemos uso de sus propios valores previamente computados para argumentos menores que *y*; pudiendo emplearse también funciones previamente definidas (Soare 1996, 1999, 2009; véase también Gödel 1931; Kleene 1952, 2002; Cutland 1980). Éste es el sentido que constituye su significado original.

Epstein y Carnielli (1989) indican que, en su forma más simple, la definición recursiva de una función *f* sería como sigue (siendo *g* una función previamente definida): $f(0) = m$; $f(n+1) = g(f(n))$. Este sistema de ecuaciones recursivas puede instanciarse fácilmente y de una manera perspicua en la función suma (cf. Boolos y Jeffrey 1974).

$a+0 = a$ /a+1 = a′ [caso base],
$a+(b+1) = (a+b)+1$ [paso recursivo].

Aunque la función suma se defina recursivamente, no tiene que proceder (operar o implementarse, tanto de manera abstracta como a tiempo real) necesariamente de manera recursiva (cf. Abelson, et al., 1996). Una implementación recursiva deriva de una definición

1 Así, las funciones no son intrínsecamente recursivas, sino que lo son en virtud de cómo se definen. Como muestro más abajo, y este es un punto crucial, una definición tal es una regla gramatical en el sentido establecido por Wittgenstein (Mota 2013).

recursiva como la que ofrezco en este trabajo, y un ejemplo del primer caso sería el siguiente: si se quiere computar 2+2, el procedimiento invoca un valor previamente computado para un argumento menor, esto es, se computaría (2+1)+1; con lo que obtenemos '2+2 = (2+1)+1 = ((2+0)+1)+1'; pero una implementación iterativa de '2+2' puede ser como sigue: se obtiene el sucesor de 1 (1+1) y al resultado (2) se le añade 1, y al resultado (3) se le añade 1, hasta que se alcanza el valor '4' (lo que se puede expresar como 2+2 = 1+1+1+1). Mediante tal implementación iterativa, una operación genera sucesivamente valores desde un argumento a otro sin usar valores previamente calculados para argumentos menores. Como muestro más abajo, tanto a la definición recursiva como a la implementación (recursiva e iterativa) subyacen reglas gramaticales en el sentido establecido por Wittgenstein.

La función suma es una función que pertenece a la clase de las funciones recursivas primitivas, empleadas y definidas por Skolem (1923) y Gödel (1931, 1934); siendo Gödel quien, además, amplió la clase de las funciones recursivas primitivas formulando la clase de las funciones recursivas generales en 1934; éstas últimas, por tanto, incluyen a las primeras puesto que no agotaban todas las funciones totales que podían definirse por recursión. Una diferencia fundamental entre la clase de las funciones recursivas generales y la clase de las funciones recursivas primitivas es que en las primeras la recursión se aplica sobre, al menos, dos variables simultáneamente (véase Gödel 1934, 69).[2]

La clase de las funciones recursivas parciales fueron definidas por Kleene (1938, 1943, 1952), e identifica correctamente la clase de las funciones computables (Kleene 1952, 323-348). La clase definida por Kleene recibe su nombre por el hecho de que una función no necesita estar definida para todas las n-tuplas de números naturales que toma como argumentos. Asimismo incluye a las funciones recursivas primitivas y a las funciones recursivas generales como aquellas funciones parciales para las cuales está definido todo el conjunto de

2 Un ejemplo de función recursiva general es: $\varphi(0,y) = \psi(y)$ Def., $\varphi(x+1, 0) = \chi(x)$ Def., $\varphi(x+1, y+1) = \varphi(x, \varphi(x+1, y))$ Def.

sus argumentos (esto es, como funciones recursivas totales).[3] Estas clases diferentes de funciones recursivas surgieron como respuesta directa o indirecta a los problemas planteados por Hilbert, los cuales incidían directamente en los fundamentos de la matemática.[4] Así, con la aparición del programa finitista de Hilbert y con la formulación de su conocido 'problema de la decisión' (*Entscheidungsproblem*), se convirtió en tarea prioritaria definir formalmente la noción de procedimiento mecánico finito (un procedimiento de decisión o *Entscheidungsverfahren*) que permitiera determinar o decidir, en un número finito de operaciones o pasos, si una expresión (o fórmula) bien formada es o no válida. Tales formulaciones constituyeron una aproximación a la noción de computabilidad, representando tales clases diversos intentos por identificar la clase de las funciones computables o calculables. Sin embargo, tales definiciones formales no fueron las únicas soluciones dadas.

Por su parte, Church (1932) formuló el cálculo-λ y posteriormente demostró que dicho formalismo era extensionalmente equivalente a las funciones recursivas generales, lo que le llevó a formular la famosa Tesis de Church, que establece que toda función efectivamente computable es

---

3 Véase Mota (2013) para ver una definición explícita de la *Tesis de Kleene* y la *Tesis de Turing-Kleene* (que recoge la equivalencia extensional indicada por Epstein y Carbielli 1989) que establece que *una función es computable si y sólo si es una función recursiva parcial (o está definida por el formalismo de Kleene) o, de forma equivalente, si es computable/calculable por una máquina de Turing;* que bien podría substituir a la *Tesis de Church-Turing* dado que, como señalo en este artículo fue Kleene, y no Church, quien identificó correctamente la clase de las funciones computables. Por otro lado, aunque el formalismo de Kleene y el de Turing puedan ser definidos por recursión, ambos proceden (operan o pueden implementarse) de manera diferente; pudiendo ser en el primer caso recursivamente, mientras que en el segundo es iterativamente. Nótese que la definición por recursión y la implementación pertenecen a niveles de análisis diferentes (sobre *qué* y *cómo*, respectivamente) y conviene no confundirlos, como argumento más abajo.

4 El *formalismo*, encabezado por Hilbert, es una concepción no-descriptivista de la matemática, enfrentada a perspectivas descriptivstas como los realistas, que sostenían la existencia real de los objetos matemáticos. Éstos últimos sostenían que las ecuaciones matemáticas eran proposiciones descriptivas de una realidad (platónica o empírica). Como muestro, Wittgenstein tampoco sostuvo una visión realista/descriptivista de la matemática ni de la lógica, pues sostenía que los objetos lógico-matemáticos eran construcciones o invenciones constituidas por reglas gramaticales (cf. Mota, 2013).

recursiva general (Church 1936).[5] Turing también formalizó la noción de computación mediante el formalismo conocido como la máquina de Turing (1937).[6] En su caso, la Tesis de Turing, asevera que una función es computable si y sólo si es computable por una máquina de Turing (Soare 2009, 373).

Uno de los autores más reseñables que recibió la obra de Turing fue Post (1921, 1943, 1944), cuyo formalismo consiste en un sistema capaz de generar todas las proposiciones de la lógica de enunciados. Así, sea *g* un enunciado de la lógica proposicional y *P* una variable operacional aplicada a dicho enunciado, el sistema produce un enunciado *g'* que sustituye a *g*. Esto se expresa en su sistema de producción normal como sigue: *gP→Pg'* (Post 1943, 199). Su tesis, conocida como la Tesis de Post, establece que un conjunto no vacío (como el de las proposiciones de las lógica de enunciados) es efectivamente numerable si y sólo si es derivado de un sistema de producción (normal) [derivado de su sistema canónico (normal)] (Davis 1982; Soare 2009).[7]

---

5 Sin embargo, parece que la identificación hecha por Church de las funciones computables con las funciones recursivas generales adolecía de errores (véase Soare 2009).

6 Turing (1937) presentó un sistema formal que consistía en una máquina de carácter abstracto equipada con cinta potencialmente infinita dividida en cuadrados, cada uno de los cuales podía expresar un símbolo. Asimismo, la máquina cuenta con una cabeza lectora que escanea un único cuadrado de la cinta cada vez. De forma muy esquemática, dicha máquina cuenta con una serie finita de condiciones ($q_1 \ldots q_n$), que son las distintas configuraciones de la máquina, una serie de símbolos (por ejemplo, 0 y 1) y, un conjunto de operaciones (como por ejemplo, escribir un nuevo símbolo; borrar un símbolo escrito; o mover su cabeza lectora un cuadrado hacia la izquierda o hacia la derecha).Es frecuente expresar las instrucciones de la máquina aludiendo a su estado inicial, el símbolo que está escaneando, el estado siguiente y la operación a realizar. Esta cuádrupla se suele expresar de la siguiente forma: E_0, 0, E_1, 1. Lo cual significa que la máquina está en el estado E_0, escaneando el símbolo '0' y pasa al estado E_1 al sustituir '0' por '1'. Esta manera de proceder es iterativa dado que se llevan a cabo diferentes operaciones de manera sucesiva, pasando así de una configuración a otra (esto es, de un estado a otro en función del símbolo escaneado y de la operación a realizar).

7 Post (1944) también se centró en los conjuntos recursivamente numerables, distinguiendo entre éstos y los conjuntos recursivos. Un conjunto *S* es recursivo si se puede establecer un método efectivo –un procedimiento mecánico finito– para determinar si, por ejemplo, un número entero positivo *n* pertenece o no al conjunto *S*. Un conjunto *S* es recesivamente numerable si existe un procedimiento mecánico efectivo para numerar efectivamente los elementos de *S*.

Como indiqué al principio de este apartado, el concepto de recursión se ha aplicado con cierta ambigüedad dentro de la teoría de la computabilidad. Así, tal y como señala Soare (1996, 1999, 2009), el concepto de recursión se comenzó a aplicar para significar 'computabilidad'. Sin embargo, el término 'computabilidad' (o 'efectivamente computable') se refiere a una función para la que se establece un procedimiento (efectivo) mecánico finito (procedimiento generativo/computacional o algoritmo) por medio del cual es computada o calculada; esto es, si se ha definido para ella una secuencia de reglas con las que, dado un input, se obtuviera un output o valor, a través de un número finito de pasos. Desde 1996, Soare ha propuesto volver a emplear el concepto de recursión en su sentido original, distinguiéndolo del de computabilidad, un concepto relacionado pero no coextensivo, dado que hay formalismos que no proceden recursivamente, como la máquina de Turing.

Como puede apreciarse, una rápida mirada sobre la historia y evolución del concepto de recursión en su ámbito de aplicación original muestra que no parece haber un lugar destacado para Wittgenstein, algo que, como veremos, no representa la verdadera aportación de este filósofo a la Teoría de la Computabilidad.

2. Wittgenstein y la definición recursiva

Con respecto a la relación entre Wittgenstein y la Teoría de la Computabilidad, son obras de interés las de Marion (1995, 1998), Frascolla (1994) y Odifreddi (2001), los cuales indican, por ejemplo, las relaciones entre algunos planteamientos de Church (1932, 1936) y Wittgenstein (1922).[8]

8 Wittgenstein estableció en 6.02 la forma general de un número natural mediante la expresión '$[x, \xi, \Omega'\xi]$' y en 6.021 mediante esta otra '$[\Omega^{0'}x, \Omega^{v'}x, \Omega^{v+1'}x]$'. Wittgenstein y Church presentaron una formulación muy parecida para generar la secuencia de los números naturales. Así, Wittgenstein aplica la variable operacional Ω n veces sobre x, lo que da lugar a la serie: x, $\Omega(x)$, $\Omega(\Omega(x))$,… (véase 6.02 para las series propuestas). Por su parte, Church definió la serie de los números naturales bajo la formulación del cálculo-λ. De forma muy breve, el cálculo-λ está basado principalmente en las operaciones de aplicación (una operación F se aplica sobre un argumento X) y de abstracción (que genera fórmulas tales como λxM, o más explícitamente, $(\lambda x.M[X])N$ donde x es una variable en la función $M[X]$ que toma a N como argumento: $(\lambda x.M[X])N = M[N]$ (veamos un

También son destacables los trabajos de Rodych (1999, 2002, 2003) en torno a la relación entre los desarrollos de Gödel y Wittgenstein. Otras obras que indican el conocimiento que Wittgenstein tenía sobre los avances en la Teoría de la Computabilidad son Wrigley (1977), Shanker (1987) y los apuntes que el propio Wittgenstein hizo sobre, por ejemplo, el trabajo de Skolem y las críticas al programa hilbertiano (Wittgenstein 1974, 1975a).

Sin embargo, ninguno de ellos parece presentar de forma explícita el papel de la recursión en la obra de Wittgenstein, presente ya desde el *Tractatus* hasta su formulación posterior centrada en la noción de regla gramatical (véase Mota 2013 para un ejemplo de tal relación).

Wittgenstein distingue entre los niveles de análisis en torno a *qué* hace un procedimiento mecánico finito (su definición formal) y *cómo* lo hace (cómo procede u opera), algo que se aprecia en el *Tractatus* (1922); siendo el primer nivel el que se aplica a la definición de la forma general y el segundo a la aplicación de la variable operacional, que caracteriza cómo genera los distintos términos de la serie (cf. Wittgenstein 1975a, §154).

La primera forma general que define Wittgenstein, y de la que se derivan las demás es la forma general de una serie de formas. Wittgenstein expresa tal forma como sigue: [*a*, *x*, *O'x*]. En 5.2522, Wittgenstein explica en qué consiste esta notación. Así, "[e]l primer término…es el comienzo de la serie de formas, el segundo es la forma de un término *x* cualquiera de la serie y el tercero la forma de aquel término de la serie que sigue inmediatamente a *x*".

Este procedimiento genera series como la siguiente: *a*, *O'a*, *O'O'a*, y así sucesivamente. La operación '*O*' se aplica iterativamente, esto es, de manera sucesiva, sobre su último resultado producido.[9] Pero, ¿Qué

---

ejemplo: $(\lambda x.2+x+x)1 = 2+1+1 = 4$). La serie de los números naturales puede generarse, así, mediante la aplicación del operador *F* sobre *X*. Así, la definición recursiva para ambos procedimientos es como sigue: $\Omega^{0'}x = x$ (en el caso del formalismo de Church: $F^{(0)}(X) = X$); $\Omega^{v+1'}x = \Omega'\Omega^{v'}x$ (en el caso del formalismo de Church: $F^{(n+1)}(X) = F(F^{(n)}(X))$).

9 Un ejemplo es: Así, '*O'O'O'O'a*' es el resultado de cuatro operaciones sucesivas (iterativas) sobre *a*, $O^{\gamma(4)}(a)$, de tal manera que la primera aplicación sobre *a*, da como resultado *O'a* [= $O^{\gamma(1)}(a)$] otra aplicación da como resultado *O'O'a* [= $O^{\gamma(2)}(a)$], otra aplicación de *O'* sobre su último resultado es *O'O'O'a* [= $O^{\gamma(3)}(a)$] y la última aplicación de *O'* da como resultado *O'O'O'O'a* [= $O^{\gamma(4)}(a)$] (cf. 5.2521).

entiende Wittgenstein por operación? Para Wittgenstein, una operación "muestra cómo se puede pasar de una forma de proposición a otra" (5.24). Dado que una serie de formas "es una serie que está ordenada por relaciones internas" (4.1252) y "[l]as estructuras de las proposiciones están en relaciones internas entre sí" (5.2), se puede concluir que una serie de proposiciones es una serie de formas.[10] La serie de los números naturales también es una serie de formas, tal y como señala Wittgenstein en 4.1252. En este sentido, "[l]a relación interna que ordena una serie equivale a la operación mediante la que un término resulta a partir de otro" (5.232), siendo posible sólo de esta manera "la progresión de un término a otro en una serie de formas" (5.252).

Dicho esto, podemos aplicar ahora los dos niveles anteriormente citados sobre la generación de una serie de formas. Así, podemos decir que lo que hace este procedimiento es definir/generar recursivamente términos de la serie. De este modo, la serie (y cada término) está *definida por recursión* en el sentido técnico arriba precisado. Por su parte, cómo hace el procedimiento para generar tales términos queda explicado mediante la aplicación sucesiva (iterativa) de la operación *O'* sobre el último elemento generado en cada paso. Formalmente, este procedimiento puede definirse recursivamente como sigue (Mota 2013, §2); siendo la primera ecuación el caso base y la segunda ecuación el paso recursivo:

Def.
$O^{(0)}{}'(a) = a,$
$O^{(n)}{}'(a) = O'O^{(n-1)}{}'(a).$

Este análisis se aplica exactamente igual al procedimiento mecánico finito que queda expresado mediante la forma general de una proposición que Wittgenstein presenta en 6: $[\bar{p}, \bar{\xi}, N(\bar{\xi})]$. El primer término de la expresión se refiere al conjunto de las proposiciones base o elementales, el segundo término se refiere a una variable proposicional (véase 5.501) y el tercer término es aquel que se obtiene al aplicar

10 De hecho, Wittgenstein establece que las funciones de verdad pueden ordenarse en series (5.1).

la operación $N(\ )$ a $\bar{\xi}$ (véase por ejemplo, 5.502 y 5.51).[11] De este modo, por medio de la negación conjunta, podemos hacer surgir una nueva proposición $N(\bar{\xi})$ a partir de ξ, previamente generada. Así, "[l]a operación es lo que le ha de suceder a una proposición para hacer surgir otra distinta a partir de ella" (5.23), y como el propio Wittgenstein se encarga de establecer, "[e]sto no dice sino que toda proposición es un resultado de aplicaciones sucesivas [iterativas] de la operación $N(\bar{\xi})$ a proposiciones elementales" (6.001; véase también 6.002).

Dado que la serie de las proposiciones es una serie de formas (y por tanto los niveles de análisis arriba señalados se aplican de igual manera), tal forma general puede definirse recursivamente como sigue (Mota 2013, §2):

Def.
$N^{(0)}{}'(p,q) = p,q,$
$N^{(n)}{}'(p,q) = N'N^{(n-1)}{}'(p,q).$[12]

Por último, Wittgenstein también proporciona un procedimiento mecánico finito que permite definir/generar la serie de los números naturales; serie sobre la que se sustenta toda la aritmética. Así, la forma general de un número natural (o entero positivo) es '[0, ξ, ξ + 1]'. Lo que hace el presente procedimiento es generar recursivamente la serie de los números naturales: I, (I)+I, ((I)+I)+I, (((I)+I)+I)+I, y así sucesivamente (Wittgenstein 1978); aplicando sucesivamente (iterativamente) la operación '+1'. Su definición formal, al ser una serie de formas, es igual que la que he presentado en relación con la forma general de una serie de formas.[13]

---

11 Como el propio Wittgenstein señala en 5.502, "$N(\bar{\xi})$ es la negación de todos los valores de la variable proposicional ξ". Por tanto, como indica en 5.51, si $\xi = p$, entonces $N(\bar{\xi})$ = (FVFV) (p). Por otra parte, si $\xi = p,q$, entonces $N(\bar{\xi})$ = (FFFV) (p,q). Así, la operación $N(\bar{\xi})$ es equivalente a la operación de la negación conjunta, la cual se representa por medio de la barra de Sheffer (Mounce, 1981).

12 Nótese que esta definición se aplica de igual manera para $\xi = p$.

13 Es crucial, siguiendo mi argumentación, entender que no hay nada intrínsecamente recursivo en una expresión como '(((I)+I)+I)+I', dado que (((I)+I)+I)+I = I+I+I+I o (((I)+I)+I)+I = (I+I)+(I+I), dos definiciones no recursivas. La recursión no es en absoluto una propiedad de las expresiones aisladas, sino de la regla: $\Omega^{(n)}{}'\mathrm{I} = \Omega'\Omega^{(n-1)}{}'\mathrm{I}$.

Es posible generalizar la formulación de la forma general de una serie de formas a la forma general de un procedimiento mecánico finito. Tal forma general expresa lo que es común a todos ellos como sistemas matemáticos formales (expresa, por tanto, una variable). Así, todos los procedimientos mecánicos finitos presentados en la sección anterior y en la actual constan de un conjunto de inputs base ($\sigma\tilde{}$), una variable que puede adoptar el valor de un subconjunto de dichos inputs ($\varepsilon\tilde{}$) y una variable operacional ($\tilde{O}(\ )$) que genera nuevos valores ($\tilde{O}(\varepsilon\tilde{})$) haciendo uso de valores previamente computados ($\varepsilon\tilde{}$). Esta formulación puede resumirse en al siguiente expresión: [$\sigma\tilde{}$, $\varepsilon\tilde{}$, $\tilde{O}(\varepsilon\tilde{})$] (Mota 2013, §2).

Así, sean los inputs base números naturales o enunciados de la lógica proposicional, y adopte la variable operacional $\tilde{O}(\ )$ el valor que adopte (por ejemplo, la operación de sumar, como en el caso de las funciones recursivas; los operadores de la lógica proposicional, como en el caso de los formalismos de Wittgenstein o Post; o las operaciones que lleva a cabo una máquina de Turing) se pueden definir para todos ellos la siguiente serie: $\varepsilon\tilde{}$, $\tilde{O}\varepsilon\tilde{}$, $\tilde{O}\tilde{O}\varepsilon\tilde{}$… *y así sucesivamente*. La iteración caracteriza cómo se genera cada término mediante la aplicación iterativa de la variable operacional en cada paso, pero cada uno de sus términos y, por tanto, toda la serie, se *define por recursión* (*íbid*.):

Def.
$\tilde{O}^{(0)}{}'(\varepsilon\tilde{}) = \varepsilon\tilde{}$,
$\tilde{O}^{(n)}{}'(\varepsilon\tilde{}) = \tilde{O}'\tilde{O}^{(n-1)}{}'(\varepsilon\tilde{})$.

Como señaló Wittgenstein (1975a, § 154), un sistema formal, como lo son todos y cada uno de los sistemas arriba mencionados, es una serie de formas, esto es, cada uno de sus términos están ordenados por relaciones internas, tal y como acabo de explicar.

Veamos varios ejemplos. En el caso de las funciones recursivas, podemos ilustrar lo dicho mediante la función suma. Un ejemplo concreto de la serie formal '$\varepsilon\tilde{}$, $\tilde{O}\varepsilon\tilde{}$, $\tilde{O}\tilde{O}\varepsilon\tilde{}$…' puede ser '2+2, 2+2+1, 2+2+1+1…'. Tal serie puede definirse recursivamente mediante el siguiente sistema de ecuaciones recursivas:

$\tilde{O}^{(0)}$'(2,2) = 2,2 (=[(2+2) = (2+2)]),
$\tilde{O}^{(2)}$'(2,2) = $\tilde{O}$'$\tilde{O}^{(1)}$'(2,2) (=[(2+4) = (2+3)+1]).

Tal sistema puede expresarse en el formalismo de Wittgenstein, aquí reproducido como [Ω^{0}'x, $\Omega^{v\text{-}1}$'x, Ω^{v}'x]. Así pues, sea la serie de definiciones $\Omega^{(2+2)}$'x, $\Omega^{(2+3)}$'x, $\Omega^{(2+4)}$'x vemos que:

$\Omega^{(2+2)}$'x = Ω^{2}'Ω^{2}'x (=[Ω^{0}'x = x]),
$\Omega^{(2+4)}$'x = Ω'$\Omega^{(2+3)}$'x (=[Ω^{v}'x = Ω'$\Omega^{v\text{-}1}$'x]).

Pero esta misma forma general, puede aplicarse con la misma corrección al formalismo de Post, el cual es capaz de generar todas las proposiciones de la lógica de enunciados. Recordemos que si a *g* (un enunciado de la lógica proposicional) se le aplica *P* (una variable operacional) el sistema produce un enunciado *g'* que sustituye a *g*. Así, podemos decir que lo que hace tal procedimiento es definir recursivamente las proposiciones de la lógica de enunciados como sigue:

Def.
$P^{(0)}$'(g) = g,
$P^{(n)}$'(g) = P'$P^{(n\text{-}1)}$'(g).

Así, vemos cómo las distintas proposiciones de la lógica de enunciados se generan mediante sucesivas aplicaciones de la variable operacional *P* sobre *g*. Por otro lado, los paralelismos entre este formalismo y el presentado por Wittgenstein en su *Tractatus*, presentado unas líneas más arriba, resulta bastante claro.

Por otro lado, la forma general arriba definida también es aplicable al formalismo de Turing. Así, partiendo de un input base, por ejemplo, el símbolo '0', este formalismo genera, aplicando las diferentes operaciones arriba señaladas (ver nota al pie 6), los distintos términos de la serie. Tal serie es igual que la que he presentado arriba: 0, $\tilde{O}$(0), $\tilde{O}\tilde{O}$(0),…y así sucesivamente. Veámoslo en un diagrama de estados:

… 0 0 0 0 … (esta es la cinta simplificada)
a. La primera instrucción es: E_0, 0, E_1, 1: la máquina, estando en E_0

escaneando '0', pasa al estado E_1 substituyendo '0' por '1':
… 1 0 0 0 … *[*E_1 se define como $\tilde{O}$ sobre 0*]*

a. En este segundo paso, las instrucciones son: E_1, 1, E_2, >: la máquina, estando en E_1 escaneando '1', pasa a E_2 moviéndose a la derecha:
[E_2 se define como $\tilde{O}$ sobre E_1 o como $\tilde{O}$ sobre $\tilde{O}(0)$*]*
b. En el tercer paso, las instrucciones son: E_2, 0, E_3, 1: la máquina, estando en E_2, escaneando '0', pasa a E_3 substituyendo '0' por '1':
… 1 1 0 0 …*[*E_3 se define como $\tilde{O}$ sobre E_2, o como $\tilde{O}$ sobre $\tilde{O}'\tilde{O}(0)$*]*

(y así sucesivamente hasta que la máquina se detiene; omito aquí ese paso)

La formalización que presentó Turing no estaba expresada mediante sistemas de ecuaciones, pero sería posible hacerlo, tal y como muestro a continuación:

Def.
$\tilde{O}^{(0)}{}'(0) = 0,$
$\tilde{O}^{(n)}{}'(0) = \tilde{O}'\tilde{O}^{(n-1)}{}'(0).$

Vemos así que la primera ecuación expresa el esto E_0 de la máquina, mientras que la segunda ecuación expresa la definición recursiva de los distintos términos/estados. Es importante señalar que '$\tilde{O}$' es una variable operacional cuyos valores son las distintas operaciones que realiza la máquina de Turing. Esto sólo indica que la serie generada por una máquina de Turing puede definirse recursivamente y esto es correcto en el nivel de análisis relacionado con *qué* hace tal procedimiento. Sin embargo, respecto a *cómo* procede u opera es claro que lo hace iterativamente, pasando en sucesión de un estado a otro.

Esta forma general de un procedimiento mecánico finito deriva, pues, de la forma general de una serie de formas, definida en el *Tractatus*.[14] Como

14 Las distintas formulaciones de los diferentes procedimientos no son sino instancias del concepto formal de procedimiento mecánico finito definido en el *Tractatus Logico-Philosophicus* por Wittgenstein en términos generales de una serie de formas. Así, *cómo* operan o proceden los distintos procedimientos mecánicos finitos (iterativamente una máquina de Turing o recursivamente una función recursiva) no es esencial a dicho

he mostrado, tal forma general puede definirse recursivamente. Pero una definición recursiva del tipo '$a+b' = (a+b)+1$' o '$\tilde{O}^{(n)}$'(0) = $\tilde{O}$'$\tilde{O}^{(n-1)}$'(0)' no expresa sino una regla gramatical (Mota 2013). Es turno, por tanto, de analizar en qué consisten las reglas gramaticales.[15] Siguiendo a Wittgenstein en las *Observaciones Filosóficas* (1975a, § 7), vemos que las reglas gramaticales "no permiten que se les justifique mediante la descripción de lo representado. Toda descripción así presupone ya las reglas de la gramática". Es decir, una regla no es gramatical si puede ser justificada "por el hecho de que una representación acorde con ella coincida con la realidad" (Wittgenstein 1974, §134). Por tanto, un criterio claro y seguro para hablar de regla gramatical es su imposibilidad de justificación mediante una realidad (empírica o platónica) independiente de la regla. Por esta razón, Wittgenstein (1974, § 133), indica que la reglas gramaticales "no tienen que rendirle cuentas a ninguna realidad". Por ejemplo, no puede haber discusión sobre si las reglas para una palabra cualquiera son correctas o adecuadas dado que las reglas no dependen de ningún significado (éste no es una realidad independiente de la regla), lo constituye; toda discusión en torno a su significado presupone ya la regla, pues sin ella la palabra no tendría significado. Otro ejemplo muy claro con respecto a la imposibilidad de acudir a una realidad independiente para justificar una regla (gramatical) lo encontramos en la *Gramática Filosófica* (§ 133), cuando Wittgenstein indica que aquellas son arbitrarias al igual que una unidad de medida en el sentido de que la elección de una unidad de medida (que no es más que un sistema de reglas del tipo '1m = 1000mm') es independiente de las propiedades físicas del objeto de la

concepto formal. De este modo, las diferencias intensionales son válidas y muy relevantes para identificar, por ejemplo, cuáles pueden ser las formulaciones de un procedimiento más claras y perspicuas, pero no para formular la forma general (el concepto formal) de un procedimiento mecánico finito, puesto que todas ellas caen bajo dicha forma general. La razón fundamental es porque lo esencial a todo procedimiento es lo que todos ellos tienen en común entre sí. Lo esencial es, así, lo recogido en la forma general (Mota 2013, §2).

15 La noción de regla gramatical comienza a aparecer en las *Observaciones Filosóficas*, un texto que contiene las reflexiones de Wittgenstein a su llegada a Cambridge en 1929. Es una noción que no aparece en el *Tractatus* explícitamente pero que posteriormente Wittgenstein iba a mostrar que estaba en la base de formulaciones tan importantes como la forma/término general de una serie de formas.

medida. Por tanto, un sistema de reglas gramaticales (como el octaedro para los colores) es independiente de las condiciones en las que se puede tener una post-imagen roja, lo cual puede establecerse mediante un experimento, pero lo primero no; la gramática es *a priori* (Wittgenstein 1975a, § 1); y lo es precisamente porque no hay una realidad independiente que pueda servir para justificar la gramática, pues sin ella, las palabras para los colores, por ejemplo, no tendrían aún significado. Toda descripción y relación entre colores presupone ya la gramática.[16]

En el dominio de la matemática, el propio Wittgenstein (1975b, 48) indica que expresiones matemáticas como '2+2 = 4' son reglas. Concretamente, son reglas que en enunciados empíricos permiten poner '2+2' en lugar de '4' (cf. Wittgenstein 1975b, 82). No hay duda, por tanto, que las ecuaciones matemáticas son reglas gramaticales, dado que el intercambio entre expresiones no depende de las propiedades de los objetos que aparecen en enunciados empíricos del tipo 'dos piedras + dos piedras = cuatro piedras'. No se puede acudir a una realidad independiente de la regla, como pueda ser la realidad empírica (obviamente, tampoco sería adecuado acudir a una realidad platónica) para justificar la regla, pues el intercambio entre expresiones presupone ya la regla.[17] Por tanto, ver que '2+2' es intercambiable con '4' no depende de ninguna comparación con los hechos (cf. 6.2321). Por otro lado, conviene indicar que Wittgenstein expresó claramente en su *Tractatus* que "las proposiciones de la matemática no expresan pensamiento alguno" (6.21). Así, las ecuaciones matemáticas, como las proposiciones de la lógica, carecen de sentido, no dicen nada, no son una figura ni una descripción de una realidad independiente. Mounce (1981, 63) indica algo parecido: "no dicen nada ni sobre el mundo ni *sobre su propia forma*". La regla muestra (no dice) que podemos substituir '2+2' por '4' constituyendo, de este modo, el significado de ambas

16 Las reglas no gramaticales o no convencionales como las reglas de cocina sí pueden justificarse por medio de una realidad independiente de la regla. Así, una regla es correcta si al seguirla pretendemos obtener un buen sabor (Wittgenstein 1974).

17 Este intercambio o substitución entre expresiones es "el método que utiliza la matemática para llegar a sus ecuaciones […]"; "las ecuaciones expresan la substitutibilidad de dos expresiones; y pasamos de cierto número de ecuaciones a ecuaciones nuevas al substituir unas expresiones por otras de una manera que responda a las ecuaciones." (6.24)

expresiones. Así, "la aritmética es la gramática de los números" (Wittgenstein 1975a, § 108) y su representación perspicua mostrará propiedades relevantes de las ecuaciones a las que subyace.

Una de tales reglas es la definición recursiva, expresada como 'a+b' = (a+b)+1'. Una definición tal, es, como el propio Wittgenstein (1975a, §163) señala en las *Observaciones Filosóficas*, una regla fundamental del sistema que nos indica cómo puedo proceder (y como tal, no se puede aseverar o negar). Del mismo modo, en la *Gramática Filosófica*, nos indica que "es una regla para la construcción de reglas de sustitución, o también el término general de una serie de definiciones [formas]" (1974, 36, 851).[18]

Como acabo de mostrar, no hay una realidad independiente que pueda servir para justificar la regla gramatical. De forma similar, la propiedad de la recursión no se puede justificar acudiendo únicamente a la organización interna de las estructuras gramaticales (véase la siguiente sección) o de datos (como '(a+b)+c'). Así, la ecuación "2+2 = (2+1)+1" expresa una regla recursiva que muestra que se puede intercambiar '2+2' con '(2+1)+1'.[19] Las ecuaciones matemáticas expresan, así, reglas gramaticales en el sentido en que las emplea Wittgenstein, y constituyen propiedades como la recursión. Podemos decir, por tanto, que un subgrupo propio de tales reglas gramaticales son las reglas recursivas, fundamento de la propiedad de la recursión del cálculo que constituyen (Mota 2013).

Sorprende, pues, ver que en algunos artículos que se ocupan explícitamente de la historia y evolución del concepto de recursión no aparezca Wittgenstein como uno de los autores con aportaciones significativas a la Teoría de la Computabilidad (por ejemplo, uno de los autores que

---

18 Esta cita es de crucial importancia para entender la continuidad de pensamiento entre los primeros trabajos de Wittgenstein y sus posteriores escritos, teniendo a la propiedad de la recursión como eje central. Así, tal y como he mostrado, la forma general de una serie de formas, por ejemplo, '[0, ξ, ξ + 1]', expresa una regla (gramatical) del tipo '(a+ (ξ + 1)) = (a+ ξ)+1'. El caso base sería '0+a = a'; siendo el rango de a todo número natural considerando el 0 como tal.

19 Nótese que la organización interna de '((2+2)+1)+1' no justifica la recursión, dado que una regla como '3+3 = ((2+2)+1)+1', aunque similar, no es una regla recursiva. La operación se llama a sí misma, pero no se acude a valores previamente computados para argumentos menores (3+2, 3+1, 3+0) del modo especificado por una regla recursiva.

omite tales observaciones es Soare 1996, 1999, 2009). Sin embargo, tal y como muestro aquí y en trabajos anteriores (Mota 2013), Wittgenstein realiza una aportación más que reconocible a la historia y evolución de la noción de recursión, y por tanto, a la Teoría de la Computabilidad y la Lógica Matemática.

En el siguiente apartado presentaré algunas de las consecuencias que sus formulaciones pueden tener en otros dominios, como el de la Ciencia Cognitiva del Lenguaje, en el que la noción de recursión ha cobrado gran importancia.

3. La recursión como problema empírico

Es de sobra conocida la aportación de Wittgenstein a la Psicología y a la Ciencia Cognitiva (Wittgenstein *1958, 1980a, 1980b, 1982, 1992*; *Proudfoot & Copeland, 1994; Proudfoot, 2009; Racine & Müller, 2009; Susswein & Racine, 2009*). Sin embargo, no es frecuente encontrar una relación explícita entre los planteamientos de Wittgenstein en torno a la noción de recursión y la noción de forma general, por un lado, y otros planteamientos dentro de la Ciencia Cognitiva en los que la recursión sea una propiedad central de, por ejemplo, el procedimiento computacional (generativo) que subyace a la facultad del lenguaje, como es el caso de la formulación de Noam Chomsky (véase Mota, 2013, para un análisis de esta relación).

Así, la propiedad de la recursividad ha suscitado mucho interés en el campo de la ciencia cognitiva. Es muy conocido el ya clásico trabajo de Hauser, Chomsky y Fitch (2002) en el que proponen que la recursión es una propiedad central de la facultad del lenguaje. Estos autores toman como referencia el *Programa Minimista* previamente formulado por Chomsky (1995). Para Chomsky (1959, 1995, 2005, 2007, 2008, 2010, 2011, 2012), el lenguaje, esto es, el Lenguaje-I, puede ser entendido como un procedimiento generativo o sistema de cómputo capaz de generar una cantidad potencialmente infinita de expresiones jerárquicas internas. De este modo, la teoría de la gramática, que comprende el estudio del lenguaje así entendido, es el estudio de una clase especial de funciones recursivas dentro de la teoría de la computación (Chomsky 1959, 2012).

Sin embargo, en la Ciencia Cognitiva el concepto de recursión también se aplica a la organización interna de las estructuras que estudia. Así, una estructura recursiva se define como aquella en la cual un constituyente (o una estructura A) contiene a otro constituyente (o estructura B) del mismo tipo (Pinker y Jackendoff 2005; Moro 2008; Karlsson 2010; Kinsella 2010). Ejemplos de tales estructuras son tanto "*[la casa de [la sierra]*$_{SN}$*]*$_{SN}$ *es de piedra*" (dos SNs, cf. Karlsson, *op. cit*., p. 51), como "*[el ratón [que el gato [que el perro perseguía]o mordió]o corría]o*" (dos cláusulas de relativo, cf. *íbid*.).

Este uso del término 'recursión' añade un significado adicional que nada tiene que ver con su sentido original, a saber, el de auto-inclusión (*self-embedding*). Así, de acuerdo con Chomsky (1965), una estructura con auto-inclusión es aquella en la que el sintagma (o estructura) A esta auto-incluido en B si A está anidado en B y, además, A es un sintagma (o una estructura) del mismo tipo que B (véase que Chomsky no alude al concepto de recursión).

Por su parte, Moro (2008), muestra que todos los sintagmas, como por ejemplo, *muchas casas viejas*, presentan el esquema asimétrico siguiente: [*Especificador*-[*Núcleo*-*Complemento*]]. Así, el especificador es *muchas*, el núcleo es *casas* y el complemento es *viejas*. Como señala Moro (*op. cit*., p. 71), tanto el especificador como el complemento (que añaden diferente información sobre el núcleo) pueden presentar el mismo esquema asimétrico, que aplicado al esquema anterior sería […*Núcleo*…-[*Núcleo*-[…*Núcleo*…]]]. Este uso de recursión, sin embargo, sigue significando lo mismo que en el caso de los constituyentes, a saber: auto-inclusión –esquemas dentro de esquemas del mismo tipo–.

Emplear de manera superficial el concepto de recursión para significar 'auto-inclusión' refleja una falta de precisión conceptual y produce confusión dentro de la Ciencia Cognitiva. De hecho, aquellos trabajos que han tratado de precisar la noción de recursión dentro de la teoría sintáctica, cometen el error de creer que en cada clase de funciones recursivas se define de manera distinta la noción de recursión (tal es el caso de Tomalin, 2007, 2011). Esto es un error porque la noción de recursión es invariable respecto de la cantidad de argumentos que se

definen; todas ellas son recursivas por la misma razón, esto es, porque para definir un nuevo valor se hace uso de un valor/valores previamente computado/s para argumentos menores.

Tal y como defiendo en otro trabajo (Mota 2013), Chomsky ha empleado correctamente la recursión como propiedad que define su procedimiento generativo. En ese mismo trabajo defino recursivamente de forma clara tal procedimiento, cuya operación básica es *Merge*: la forma general de *Merge* se puede formular de la siguiente manera, siguiendo la notación de Wittgenstein: [N, O_s, $M(O_s)$]. Así, N hace referencia a ítems léxicos y objetos sintácticos que configuran la numeración, O_s hace referencia a un objeto sintáctico cualquiera de la serie generada y $M(O_s)$ hace referencia a un objeto sintáctico nuevo generado a partir de O_s mediante la aplicación de M ( ). Así, sea la numeración N = {*Juan, canta, muchas, canciones*}, lo que *Merge* hace es tomar dos ítems léxicos (definida aquí como una operación binaria) 'muchas' y 'canciones' y forma el objeto sintáctico {muchas, canciones} = {X,Y}. Mediante otra aplicación, *Merge* genera un nuevo objeto sintáctico {canta, {muchas, canciones}}, formado a partir de un objeto previamente computado ({muchas, canciones}); por último, *Merge* genera {Juan, {canta, {muchas, canciones}}}. Así, y de manera esquemática, vemos que lo que *Merge* hace (esto es, generar/definir recursivamente objetos sintácticos, independientemente de su estructura interna, como muestro en la definición formal) queda expresado mediante la siguiente serie: O_s, $M'O_s$, $M'M'O_s$…; en el ejemplo: $O_{s\{muchas,\ canciones\}}$; $M_{canta}{}'Os_{\{muchas,\ canciones\}}$; $M_{Juan}{}'M_{\text{canta}}{}'Os_{\{muchas,\ canciones\}}$. Esto puede definirse recursivamente de manera formal como sigue (Mota, 2013, nota 2):

Def.
$M^{(0)}{}'(O_s) = O_s$,
$M^{(n)}{}'(O_s) = M'M^{(n-1)}{}'(O_s)$.

Por otra parte, Chomsky (2007, p. 6; 2008, p. 139) ha insistido en que el modo de operar (cómo procede, o puede implementarse de manera abstracta) de este procedimiento generativo es aplicando iterativamente la operación *Merge*. Así, la aplicación iterativa queda descrita mediante

la siguiente serie, que muestra, en cada paso, las *n* veces que *Merge* se aplica de forma sucesiva, lo cual puede generar todo tipo de expresiones (con auto-inclusión o no) del lenguaje natural: $M^{(0)}$, $M^{(1)}$, $M^{(2)}$,…, $M^{(n)}$. Vemos, pues, que las estructuras con auto-inclusión no justifican la propiedad de la recursión.

Es posible formular de la misma manera la propiedad de la recursión en su anterior caracterización (empleada en los años 60) del sistema generativo, a saber, un sistema de reglas de rescritura, basado en el formalismo de Post (Chomsky 1965). Veámoslo en el siguiente ejemplo:

Lo que hace un sistema de producción tal y como lo aplica Chomsky, es especificar recursivamente las oraciones de un lenguaje (Chomsky y Miller 1963). Esto, formalmente puede formularse como sigue: Sea *R* un conjunto de reglas, *s* una estructura generada/especificada por esas reglas y *R*(*s*) una nueva estructura generada/especificada por medio de la aplicación de *R*( ) sobre una estructura previamente generada/especificada *s*, podemos definir recursivamente tal generación/especificación como sigue:

Def.
$R^{(0)'}(s) = s$,
$R^{(n)'}(s) = R'R^{(n-1)'}(s)$.

Este sistema de ecuaciones recursivas expresa que cada nueva estructura (sea una estructura con auto-inclusión o no) se define o se especifica en función de estructuras previamente generadas. Además, una formulación tal da cuenta de la generación potencialmente infinita de expresiones lingüísticas. Esta definición recursiva guarda un gran parecido con los sistemas de producción de Post, tal como los he definido más arriba.

Ahora, conviene detenerse un momento para analizar las reglas específicas de una gramática. Dentro de la teoría de la gramática, una regla del tipo $\Sigma \rightarrow \varepsilon\Sigma$ se dice que es recursiva. Es recursiva porque el elemento que aparece al lado izquierdo reaparece en el lado derecho de la flecha (véase por ejemplo Luuk y Luuk, 2011). Si a esa regla le añadimos $\Sigma \rightarrow \varepsilon$, tenemos un sistema con dos reglas: $\Sigma \rightarrow \varepsilon\Sigma$; $\Sigma \rightarrow \varepsilon$. Un sistema así, genera recursivamente, tal y como he indicado en la definición arriba presentada, series tales como ε, εε, εεε,…, que puede expresarse como ε, *R*'ε, al

aplicar *R* sobre ε, *R'R'*ε y así sucesivamente. De este modo, obtenemos una serie de formas. Vemos pues que las dos formulaciones hechas por Chomsky (en términos de sistemas de producción y en términos de *Merge*) no son sino instancias de una forma general de una serie de formas, a las cuales subyacen reglas gramaticales wittgensteinianas.

Por otro lado, conviene señalar que no es una aseveración adecuada decir que lo que hacen las reglas de rescritura puede caracterizarse mediante la auto-inclusión (un ejemplo de tal aseveración es Fitch, 2010). Lo que sencillamente hacen es definir/generar una nueva descripción estructural desde una previamente generada (Post 1921). En este sentido, Post no parece especificar en sus obras las reglas con la propiedad de la auto-inclusión, por lo que no es adecuado relacionarlas con dicho término. Así, un sistema como Σ→εΣ; Σ→ε no expresa sino reglas de un cálculo que permite generar Σ, ε, εε, εεε, lo que equivale a Σ, *R'*Σ, *R'R'*Σ, *R'R'R'*Σ. A una serie tal subyacen, como he mostrado en el apartado anterior, reglas gramaticales wittgensteinianas. Así, para generar εεε, el sistema aplica εΣ→εεΣ (lo cual quiere decir que las expresiones de lado izquierdo se rescriben por las del lado derecho), que puede reformularse como '$\varepsilon\varepsilon\Sigma = R'(\varepsilon\Sigma)$'. Esta última formulación es una regla del tipo '$a+b' = (a+b)+1$', es decir, reglas gramaticales que constituyen un cálculo y no muestran sino que las expresiones son intercambiables. Muestran, por tanto, que un valor se computa haciendo uso de valores previamente computados y que un valor se substituye por otro, por lo que la noción de auto-inclusión no es adecuada para caracterizar el cálculo. Es en este sentido técnico en el que hay que tratar tales reglas como recursivas y no tanto en el que emplean, por ejemplo, Luuk y Luuk.

Diversos trabajos han discutido el papel central de la recursión en el dominio del lenguaje, pero desplazando el debate hacia la noción de auto-inclusión arriba mencionada. Un buen Ejemplo son los trabajos de Everett (2005, 2009) sobre la lengua Pirahã, en la que parece que están ausentes estructuras con auto-inclusión.

Quizá, el ejemplo más claro lo constituye Jackendoff (2011), cuando señala que hay dos tipos de recursión, que denomina como *recursión formal* y *recursión estructural*, respectivamente (*op. cit*., pp. 591-592). La primera de ellas se aproxima muy ligeramente a la noción de recursión en sentido original que he tratado arriba (de hecho se

aproxima mucho más a la de iteración), y especifica que un conjunto de reglas es recursivo si pueden aplicarse a su propio resultado un número ilimitado de veces a partir de un conjunto de ítems primitivos o no definidos (*op. cit*., p. 591). En cambio, la segunda noción se ajusta a la definición, que acabo de presentar, de 'estructura recursiva'. Así, Jackendoff (*op. cit*., p. 592) establece que una estructura es recursiva si pueden incluirse constituyentes dentro de otros (del mismo tipo) con una profundidad ilimitada. Después de establecer tal distinción, Jackendoff (*íbid*.) sostiene que de ambos usos de recursión, el de recursión estructural es más útil para los propósitos de la Ciencia Cognitiva. Así, en el lenguaje natural, nos dice Jackendoff, tenemos que establecer primero su dominio de recursión estructural y, si podemos hacerlo, entonces podemos concluir que las reglas que definen el dominio son formalmente recursivas. En otras palabras, según Jackendoff, podemos inferir la recursión formal a partir de la recursión estructural.

Pero la auto-inclusión no determina ni conceptual (es decir, desde un punto de vista lógico o formal) ni empíricamente la recursión, por lo que la aseveración de Jackendoff sencillamente no es válida. Así, por ejemplo, una expresión como $\varphi(a,x) = \psi(\varphi(a,x-1))$ expresa una definición recursiva en la cual no hay rastro de auto-inclusión. Como se ha señalado más arriba, esta ecuación muestra que la expresión del lado izquierdo ($\varphi(a,x)$) se puede *reemplazar/substituir/intercambiar* (no auto-incluir) por la del lado derecho ($\psi(\varphi(a,x-1))$), en donde ψ y φ son dos funciones diferentes.[20] Por otro lado, la recursión se encuentra en el hecho de que para definir un nuevo valor se hace uso de un valor previamente computado para un argumento menor, es decir, se hace uso de la definición por recursión, pero ninguna de las expresiones por separado es recursiva; sino la definición, la regla que subyace y que constituye dicha ecuación matemática (Mota 2013).

---

20 No hay que confundir el hecho de que en ambos lados del signo '=' aparece la función *f* con la definición de auto-inclusión. Así, lo único que indica lo primero es la auto-llamada, restringida y determinada por el hecho de que para definir un valor se hace uso de un valor previamente computado para un argumento menor por esa función, de ahí su presencia en ambos lados. Sin embargo, esto no significa ni que la función inserte valores dentro de valores del mismo tipo (algo que no tiene ningún sentido) ni tampoco que la noción de recursión se defina aquí en el sentido de la auto-inclusión. En este caso, la noción de recursión se emplea en su sentido técnico original.

Así, la recursión es una propiedad del cálculo, tanto si lo que se estudia es su definición formal, como en el caso de *Merge*, como si lo que se estudia es el algoritmo que caracteriza el procesamiento de, digamos, oraciones del lenguaje natural. Como muy bien indicó Kenny (1990) es una cuestión empírica determinar qué algoritmo subyace al procesamiento de tales oraciones –esto es, descubrir qué valor computa una función gramatical para unos argumentos dados; viendo, por ejemplo, el tiempo que tarda en dar una respuesta–. Ciertamente podemos averiguar mediante un experimento qué algoritmo se está aplicando en la resolución de una tarea, pero, añado, no estamos, por ello, comprobando experimentalmente sus propiedades. Recuérdese, la recursión es una propiedad de las reglas gramaticales (recursivas) que constituyen el cálculo (Mota 2013, nota 3); una regla que en enunciados empíricos permite substituir un valor previamente computado para un argumento menor por el valor que se define para el/los argumento/s actual/es; lo cual presupone ya le regla. Sin embargo, esto no significa que una regla gramatical (o las propiedades que constituye) sea susceptible de confirmación o disconfirmación empírica. Como he indicado más arriba, una regla tal, una definición recursiva, no puede negar o aseverar nada, pues no dice ni describe nada, señala cómo puedo proceder. Por tanto, la substitución constituida por una regla recursiva no depende de ninguna comparación con los hechos. La recursión no es una propiedad que pueda ser probada (confirmada/disconfirmada) experimentalmente.

Por tanto, no podemos justificar, ni inferir, la propiedad de la recursión, si la entendemos en su sentido original, a partir de la organización interna de las estructuras con auto-inclusión (X dentro de X del mismo tipo), cuyas propiedades son independientes de la regla recursiva; igual que no se puede acudir al valor X para inferir si éste ha sido definido por recursión.[21] Así, la noción de recursión

21 En el campo de la Ciencia de la Computación se entiende que expresiones como '$(x+y)+z$' o '$w\times(x\times(y\times z))$' son recursivas en el sentido de que un componente contiene a otro del mismo tipo. Este no es el sentido original de recursión. Este último ha de entenderse como una propiedad de una regla y no de una expresión aislada. Una regla recursiva constituye el punto de vista específico desde el que contemplar la igualdad de significado de dos expresiones (cf. Wittgenstein, 6.2323).

en Ciencia Cognitiva debería reservarse a su sentido original para evitar la confusión conceptual que supone identificar la auto-inclusión con la recursión.

Conclusiones

Las principales conclusiones que se derivan de este trabajo son las siguientes: (1) Wittgenstein debe considerarse como un autor central en los estudios sobre la historia y la evolución (así como la precisión) del concepto de recursión, pues presenta un análisis claro y sin ambigüedad sobre la definición recursiva.

(2) Wittgenstein es un autor central en el dominio de la Teoría de la Computabilidad y la Lógica Matemática, tanto por su discusión en torno al programa hilbertiano, como por sus propuestas, las cuales pueden recoger lo que es esencial a todo procedimiento mecánico finito.

(3) La noción de regla gramatical, tal como es formulada por Wittgenstein, es básica para entender la noción de recursión, tanto dentro de la Teoría de la Computabilidad como fuera de ella. Una comprensión de dicha noción, junto de otras que he presentado aquí, las cuales muestran, por otro lado, una continuidad de pensamiento, permitirá entender la noción de recursión en sentido técnico y evitar así una confusión conceptual que oscurece su sentido original.

Bibliografía

Abelson, H. & Sussman, G. J. with Sussman, J. (1996). *Structure and interpretation of computer programs*. Cambridge, MA: MIT Press.

Boolos, G. & Jeffrey, R. (1974). *Computability and logic*. Cambridge: Cambridge University Press.

Chomsky, N. & Miller, G. (1963). Introduction to the formal analysis of natural languages. In R.D. Luce, R.R. Bush & E. Galanter (*Eds*.), *Handbook of mathematical psychology*, *Vol. II* (pp. 269-322). New York: John Wiley.

Chomsky, N. (1959). On certain formal properties of grammars. *Information and Control*, *2*, 137-167.

Chomsky, N. (1965). *Aspects of theory of syntax*. Cambridge. MA: MIT Press

Chomsky, N. (1995). *The minimalist program*. Cambridge. MA: MIT Press.

Chomsky, N. (2005). Three factor in language design, *Linguistic Inquiry*, *36*, 1-22.

Chomsky, N. (2007). Approaching UG from below. In U. Sauerland & H. M. Gärtner (*Eds.*), *Interfaces + Recursion = Language?* (pp. 1-30). Berlin: Mouton.

Chomsky, N. (2008). On phases. In R. Freidin, C. Otero, & M. L. Zubizarreta (*Eds.*), *Foundational issues in linguistic theory* (pp. 133-166). Cambridge. MA: MIT Press.

Chomsky, N. (2010). Some simple evo devo theses: How true might they be for language? In R. Larson, V. Déprez & H. Yamakido (*Eds.*), *The evolution of human language* (pp. 45-62). Cambridge: Cambridge University Press.

Chomsky, N. (2011). Language and other cognitive systems. What is special about language?. *Language Learning and Development, 7*, 263-278.

Chomsky, N. (2012). Some core contested concepts. *Proceedings of the CUNY 2012*, 1-18.

Church, A. (1932). A set of postulates for the foundation of logic, *The Annals of Mathematics*, *33*, 346-366.

Church, A. (1936). An unsolvable problem of elementary number theory. In M. Davis (*Ed.*), *The undecidable* (pp. 88-107). New York: Raven Press.

Cutland, N. (1980). *Computability: an introduction to recursive function theory*. Cambridge: Cambridge University Press.

Davis, M. (1965). *The undecidable*. New York: Raven Press.

Davis, M. (1982). Why Gödel didn't have Church Thesis, *Information and Control*, *54*, 3-24.

Epstein, R. & Carnielli, W. (1989). *Computability: computable functions, logic, and the foundations of mathematics*. Pacific Grove, CA: Wadsworth & Brooks/Cole.

Everett, D. (2005). Cultural constraints on grammar and cognition in Pirahã, *Current Anthropology*, *46*(4), 621-646.

Everett, D. (2009). Pirahã culture and grammar: a response to some criticisms, *Language*, *85*(2), 405-442.

Fitch, T. (2010). Three meanings of recursion: key distinctions for biolinguistics. In R. Larson, V. Déprez & H. Yamakido (*Eds.*), *The evolution of human language* (pp. 73-90). Cambridge: Cambridge University Press.

Frascolla, P. (1994). *Wittgenstein's philosophy of mathematics*. London: Routledge.

Gödel, K. (1931). On formally undecidable propositions of the Principia Mathematica and related systems. I. In M. Davis (*Ed.*), *The undecidable* (pp. 4-38). New York: Raven Press.

Gödel, K. (1934). On undecidable propositions of formal mathematical systems. In M. Davis (*Ed.*), *The undecidable* (pp. 39-74). New York: Raven Press.

Hauser, M., Chomsky, N. & Fitch, T. (2002). The faculty of language: What is, who has it, and how did it evolve?, *Science*, *298*, 1569-1579.

Jackendoff, R. (2011). What is the human language faculty? Two views, *Language*, *87*, 586-624.

Janik, A. & Toulmin, S. (1996). *Wittgenstein's Vienna*. New York: Simon and Schuster.

Karlsson, F. (2010). Syntactic recursion and iteration. In H. van der Hulst (*Ed.*), *Recursion and human language* (pp. 43-67).

Kenny, A. (1990). *El legado de Wittgenstein*. Madrid: Siglo XXI

Kinsella, A. (2010). Was recursion the key step in the evolution of the human language faculty? In H. van der Hulst (*Ed.*), *Recursion and human language* (pp. 179-191).

Kleene, S. C. (1938). On notation for ordinal numbers, *The Journal of Symbolic Logic*, *3*, 150-5.

Kleene, S. C. (1943). Recursive predicates and quantifiers, *Transactions of the American Mathematical Society*, *53*, 41-73.

Kleene, S. C. (1952). *Introduction to metamathematics*. Amsterdam: North-Holland Publishing.

Kleene, S. C. (2002). *Mathematical logic*. Mineola, NY: Dover Publications.

Luuk, E. & Luuk, H. (2011). The redundancy of recursion and infinity for natural language, *Cognitive Processing*, *12*, 1-11.

Marion, M. (1995). Wittgenstein and finitism, *Synthese*, *105*, 141-176.

Marion, M. (1998). *Wittgenstein, finitism, and the foundations of mathematics*. Oxford: Oxford University Press.

Moro, A. (2008). *The boundaries of Babel*. Cambridge, MA: MIT Press.

Mota, S. (2013). La propiedad de la recursión en el "Tractatus Logico-Philosophicus" de Wittgenstein y su relación con la Teoría de la Computabilidad y la Lógica Matemática, *17*, *Observaciones Filosóficas*. 28/octubre/2013. http://www.observacionesfilosoficas.net/lapropiedaddelarecursion. htm.

Mounce, H.O. (1981). *Wittgenstein's Tractatus. An introduction*. Oxford: Blackwell.

Odifreddi, P. (2001). Recursive functions: an archeological look. In C.S. Claude, M.J. Dinneen & S. Sburlan (*Eds.*), *Combinatorics, Computability and Logic* (pp. 13-31). London: Springer-Verlag.

Pinker, S. & Jackendoff, R. (2005). The faculty of language: What's special about it?, *Cognition*, *95*, 201-236.

Post, E. (1921). Introduction to a general theory of elementary propositions, *American Journal of Mathematics*, *43*, 163-185.

Post, E. (1943). Formal reductions of the general combinatorial decision problem, *American Journal of Mathematics*, *65*, 197-215.

Post, E. (1944). Recursively enumerable sets of positive integers and their decision problems. In M. Davis (*Ed.*), *The undecidable* (pp. 305-337). New York: Raven Press.

Proudfoot, D. & Copeland, J. (1994). Turing, Wittgenstein and the science of mind, *Australian Journal of Philosophy*, *72*, 497-519.

Proudfoot, D. (2009). Meaning and mind: Wittgenstein's relevance for the 'Does Language Shape Thought?' debate, *New Ideas in Psychology*, *27*, 163-183.

Racine, T. & Müller, U. (2009). The contemporary relevance of Wittgenstein: Reflections and directions, *New Ideas in Psychology*, *27*, 107-119.

Rodych, V. (1999). Wittgenstein's inversion of Gödel's Theorem, *Erkenntnis*, *51*, 173, 206.

Rodych, V. (2002). Wittgenstein on Gödel: the newly published remarks, *Erkenntnis*, *56*, 379-397.

Rodych, V. (2003). Misunderstanding Gödel: new arguments about Wittgenstein and new remarks by Wittgenstein, *Dialectica*, *57*, 279-313.

Shanker, S.G. (1987). Wittgenstein versus Turing on nature of Church's thesis, *Notre Dame Journal of Formal Logic*, *28*, 615-649.

Skolem, T. (1923). The foundations of elementary arithmetic established by means of the recursive mode of thought, without the use of apparent variables ranging over infinite domains. In J. Van Heijenoort (*Ed.*), *From Frege to Gödel. A source book in mathematical logic, 1879-1931* (pp. 302-333). Cambridge, MA: Harvard University Press.

Soare, R. (1996). Computability and logic, *The Bulletin of Symbolic Logic*, *2*, 284-321.

Soare, R. (1999). The history and concept of computability. In E.R. Griffor (*Ed.*), *Handbook of computability theory* (pp. 3-36,). Amsterdam: North-Holland Publishing.

Soare, R. (2009). Turing oracles machines, online computing, and three displacements in computability theory, *Annals of Pure and Applied Logic*, *160*, 368-399.

Susswein, N. & Racine, T. (2009). Wittgenstein and not-just-in-the-head cognition, *New Ideas in Psychology*, 27, 184-196.

Tomalin, M. (2007). Reconsidering recursion in syntactic theory, *Lingua*, 117, 1784-1800.

Tomalin, M. (2011). Syntactic structures and recursive devices: A legacy of imprecision, Journal of Logic, *Language and Information*, 20, 297-315.

Turing, A. (1937). On computable numbers, with an application to the Entscheidungsproblem. In M. Davis (*Ed.*), *The undecidable* (pp. 116-151). New York: Raven Press.

Wittgenstein, L. (1922). *Tractatus logico-philosophicus*. London: Routledge.

Wittgenstein, L. (1958). *Philosophical investigations*. Oxford: Blackwell.

Wittgenstein, L. (1974). *Philosophical grammar*. Traducción de Luis F. Segura, ed. 1992. México, D.F.: UNAM.

Wittgenstein, L. (1975a). *Philosophical remarks*. Traducción de Alejandro Tomasini Bassols, ed. 1997. México, D.F.: UNAM.

Wittgenstein, L. (1975b). *Wittgenstein's lectures on the foundations of mathematics, Cambridge, 1939*. Chicago: University of Chicago Press.

Wittgenstein, L. (1978). *Remarks on the foundations of mathematics*. Oxford: Blackwell.

Wittgenstein, L. (1980a). *Remarks on the philosophy of Psychology. Vol. 1*. Oxford: Blackwell.

Wittgenstein, L. (1980b). *Remarks on the philosophy of psychology. Vol. 2*. Oxford: Blackwell.

Wittgenstein, L. (1982). *Last writings on the philosophy of psychology. Vol. 1*. Oxford: Blackwell.

Wittgenstein, L. (1992). *Last writings on the philosophy of psychology. Vol. 2*. Oxford: Blackwell.

Wrigley, M. (1977). Wittgenstein's philosophy of mathematics, *Philosophical Quarterly*, *27*, 50-59.