Mathesis **IV 1**$_1$ (2017) 115 – 123.

Lógica para la computación

Favio Ezequiel Miranda Perea

Luis de Ledesma. 2009. *Lógica para la computación - teorías de primer orden, resolución y elementos de programación lógica y prolog*. Coedición: Alfaomega, Ra-Ma. Páginas:180. ISBN:978-607-7854-33-3.

La lógica es una piedra angular de la metodología científica y por lo tanto pertenece a los fundamentos de cada área de conocimiento que se pueda considerar como una ciencia. Para las ciencias de la computación juega un papel central puesto que muchas de las ramas de esta aún joven disciplina emergieron de problemas y métodos desarrollados en la lógica puramente matemática. Además, esta ciencia, legado de Aristóteles, ha producido un vasto yacimiento de métodos y teorías para los fundamentos de la computación, las cuales se consideran nativas de esta área y no forman parte ya de lo que hoy entendemos como lógica matemática sino que constituyen la llamada lógica para la computación o lógica computacional. Por ejemplo de la teoría de la recursión surgió la teoría de la complejidad clásica; por otro lado, la verificación de modelos y la teoría descriptiva de la complejidad son descendientes directos de la teoría de modelos y, la teoría de la prueba tiene diversas repercusiones en la ciencia de la computación teórica; en particular la semántica de lenguajes de programación se sirve ampliamente de

los sistemas de cálculo lambda y de las teorías de tipos.

Actualmente parece imposible encontrar un área de las ciencias de la computación donde la lógica no figure directa o indirectamente. En este sentido y en analogía al papel que juega el cálculo diferencial e integral en las ciencias naturales, la lógica se ha ganado el nombre honorario de cálculo para las ciencias de la computación (véase [Halpern *et al*. 2001]). Dentro de sus aplicaciones clásicas a la computación está el objeto de estudio de este libro, la llamada programación lógica, paradigma basado en la interpretación procedimental de la lógica de predicados de primer orden, donde el proceso de cómputo coincide con la búsqueda de pruebas formales mediante cierta estrategia fija; los valores computados se obtienen mediante substituciones, llamadas unificadores más generales, las cuales son generadas automáticamente, y el control del programa lo proporciona un mecanismo de retroceso automático en el proceso de búsqueda.

Luis de Ledesma nos presenta un libro cuyo título '*Lógica para la computación*' es demasiado vago, aunque de inmediato precisa los tópicos de su interés con el subtítulo '*Teorías de primer orden, resolución y elementos de programación lógica y PROLOG*'. Es pues esta obra una monografía dedicada a ofrecer un panorama de los principales fundamentos de la programación lógica y del lenguaje de programación PROLOG. En la introducción el autor nos informa que el texto se ha pensado para ser accesible a cualquier lector inclinado a las matemáticas, la lógica y los fundamentos de su automatización y adicionalmente nos advierte que los temas expuestos distan mucho de ser originales, aunque la forma de exponerlos sí ha pretendido serlo. En efecto, los temas son clásicos pero la presentación es original dada la audiencia pensada por el autor. Precisamente por esto no encontramos el rigor matemático usual para un texto de lógica para ciencias de la computación como son [Huth y Ryan 2004] y [Spserschneider y Antoniou 2001] o de fundamentos de resolución y

programación lógica como [Lloyd 1987] y [Leitsch 1997] pero tampoco la discusión somera o nula de las cuestiones teóricas que es común en un texto dedicado a los pormenores de la programación en PROLOG, digamos [Sterling y Shapiro 1994] y [Bramer 2005]. Ambas características lo hacen atractivo para lectores con un legítimo interés en los fundamentos de la programación lógica pero que no cuentan con el bagaje matemático requerido por textos más formales, y que tampoco necesitan de un lenguaje ni de una serie de demostraciones complicadas exigidas por un público más especializado. Con esto no queremos decir que el libro sea totalmente informal o dedicado solamente a la divulgación puesto que los conceptos preliminares de la lógica matemática y los propios de la programación lógica son puntualmente explicados. En general el estilo del libro es claro y amigable, aunque en ocasiones llega a ser demasiado coloquial, por ejemplo en el caso de la definición de la jerarquía o precedencia de conectivas y cuantificadores. Sin embargo, no encontramos errores importantes debido a esta forma laxa en la presentación. Si bien, hay algunas omisiones de importancia como mencionar que en el caso de lenguajes con igualdad, además de la resolución binaria es imprescindible utilizar la regla de paramodulación para obtener un procedimiento de decisión refutacionalmente completo; o la relevancia de procesos como la subsunción y la demodulación para podar el espacio de búsqueda de la cláusula vacía, el único error por omisión encontrado en el texto resulta al no advertir al lector que, por razones de eficiencia, el algoritmo de unificación de cláusulas se implementa intensionalmente de manera incorrecta en muchos sistemas PROLOG al suprimirse la verificación de la presencia de una variable en un término.

Con respecto a las líneas de oportunidad para mejorar el texto podemos sugerir remediar la escasa presencia de ejercicios, así como ampliar la bibliografía, sobre todo en lo que respecta a citar fuentes para profundizar en cues-

tiones técnicas, como son las demostraciones ausentes en el texto y para conocer aplicaciones actuales de los métodos de la programación lógica como por ejemplo su relevancia en la llamada web semántica.

En conclusión, estamos ante una monografía adecuada para un primer acercamiento a los fundamentos de la programación lógica por parte de lectores que no requieran de la formalidad matemática de una licenciatura como la de matemáticas puras o aplicadas o la de ciencias de la computación.

Comentario a detalle

El libro se divide en tres capítulos, *teorías de primer orden, resolución y elementos de programación lógica y PROLOG*, e incluye tres apéndices, de los cuales el primero nos da una introducción a las definiciones inductivas que nos viene de la teoría de los conjuntos, puesto que se introducen las llamadas definiciones *B-F*-inductivas así como su principio de inducción. Encontramos aquí un ejemplo interesante y original de definición inductiva, a saber, la definición del conjunto de estados en la ejecución de un programa imperativo. El apéndice B es técnico y nos da una prueba inductiva del teorema de corrección del método de tablas analíticas discutido en el capítulo 1. Finalmente en el apéndice C hallamos algunas consideraciones para el proceso de especificación de información en la lógica de predicados.

Teorías de primer orden

Este capítulo abarca la mayor parte del libro y en él se discuten los conceptos básicos de la lógica de primer orden sin igualdad con base en el concepto de teoría de primer orden vista como un sistema formal, a la usanza de Shoenfield [2001]. La lógica de proposiciones se presenta como un caso particular de un lenguaje de primer orden sin variables ni cuantificadores y cuya signatura o tipo de semejanza sólo tiene símbolos de predicado de cero argumen-

tos, es decir, las variables proposicionales se identifican con los símbolos de predicado de índice cero. La sintaxis toma como primitivos a los conectivos negación y disyunción, así como al cuantificador existencial, esta elección también es tomada de Shoenfield y resulta muy adecuada puesto que el interés está en el lenguaje clausular y la programación lógica. La definición de sustitución de una variable por un término en una fórmula es la usual en lógica matemática, aunque se esperaría, siendo el libro dedicado a una audiencia más amplia, como son ingenieros en computación o informática o programadores de PROLOG, encontrar una definición recursiva más cercana al espíritu de la programación declarativa.

Entre los metateoremas y reglas derivadas discutidos, se incluyen los del cuantificador universal, el método de demostración por reducción al absurdo y el metateorema de tautología que es esencialmente el teorema de completud proposicional y permite generar teoremas mediante la noción de consecuencia tautológica. Aquí, el concepto de tautología se basa en valoraciones arbitrarias a las variables proposicionales y a las fórmulas existenciales. La sección 1.2 nos da una breve presentación del cálculo de tablas analíticas, también conocidas como tableaux semánticos, mostrándose a detalle la corrección del mismo en el apéndice B. Este apartado, aunque proporciona un método de decisión importante y más fácil de aplicar que la resolución binaria, al no requerir de la transformación a cláusulas, constituye una desviación del camino propuesto por el autor en la introducción y, de hecho, De Ledesma no nos da indicio alguno de su relación con el resto del material expuesto. La semántica formal de la lógica de predicados se presenta en la sección 1.3, donde nos encontramos con una definición de verdad que depende de la expansión del lenguaje con nombres (constantes) para todos los elementos del universo en cuestión. Siendo la de consecuencia lógica una noción tan fundamental en el

estudio semántico de la lógica y, puesto que la adecuación de la resolución binaria depende del principio de refutación de aquélla, su presentación en el texto se antoja demasiado breve; si bien en la sección 1.4 sí se discuten los grandes teoremas de la lógica como la completud y la indecidibilidad. El gran ausente aquí es el teorema de compacidad, probablemente porque en las aplicaciones computacionales sólo se lidia con conjuntos finitos de fórmulas.

La parte final del capítulo se dedica a preparar el terreno hacia la lógica clausular y el método de demostración por resolución binaria, para lo cual se discute más ampliamente la semántica usual de la lógica proposicional mediante la identificación de los valores booleanos con estructuras triviales dada la ausencia de términos y cuantificadores. Se introduce la noción de cláusula de literal, literal positivo y negativo, cláusula y cláusula de Horn. Sin embargo, se relega la muy relevante transformación de una fórmula cualquiera a un conjunto finito de cláusulas mediante la llamada forma normal conjuntiva a un ejercicio, lo cual podría provocar que el lector menos experimentado no aprecie que la lógica clausular es tan poderosa como toda la lógica en relación al análisis de consecuencias lógicas. A continuación, se presenta un algoritmo para el problema de insatisfacibilidad de un conjunto finito de cláusulas de Horn, que se denota como algoritmo SAT, cuya prueba de corrección y completud se presenta de manera clara en el texto, aunque en realidad se trata de un algoritmo de tiempo polinomial para el problema Horn-SAT. Siendo el problema SAT bien conocido, el nombre podría causar confusión en el lector poco familiarizado con la teoría de la complejidad. A este respecto observamos que, aún cuando se menciona la explosión exponencial en la complejidad del cálculo de una tabla de verdad, se extraña una mención al problema SAT y a su relevancia en la teoría de los problemas NP-completos.

Resolución

El principal cimiento del paradigma de programación lógica es la regla de resolución binaria de Robinson, discutida en este capítulo. Empezamos con una explicación de las formas normales prenex, de Skolem y conjuntiva para lograr la transformación de cualquier fórmula de la lógica de predicados en su forma clausular, es decir, en un conjunto finito de cláusulas. El mecanismo de respuesta a una meta en programación lógica se sirve de manera imprescindible del concepto de unificación de literales, al cual se le dedica un tiempo satisfactorio en la discusión, aunque el algoritmo clásico de Robinson podría sustituirse con uno más eficiente y conceptualmente más accesible como lo es el de Martelli-Montanari. Enseguida se presentan la regla de resolución binaria con unificación, los conceptos de deducción y refutación por resolución y se da una demostración del teorema de corrección, pero se echan de ménos ejemplos más elaborados y una nota sobre los algoritmos de saturación para automatizar la búsqueda de la cláusula vacía. En la siguiente sección se presentan los rudimentos de la teoría de Herbrand, necesarios para comprender la prueba del teorema de completud refutacional. Esta prueba no se presenta puesto que el libro no pretende ser riguroso en relación a las demostraciones. Sin embargo, De Ledesma nos muestra de manera excelente, mediante un ejemplo, la idea detrás de la demostración formal del teorema de completud refutacional de la resolución binaria. De esta manera, los lectores con menor conocimiento de la lógica matemática, como puede ser un programador lógico interesado en los fundamentos de su principal herramienta de trabajo, recibe una justificación adecuada de la validez lógica de las respuestas del intérprete de PROLOG.

El capítulo termina con una discusión de dos procedimientos clásicos derivados del teorema de Herbrand, el de Gilmore y el de Davis-Putnam, así como con una muy atinada nota acerca del razonamiento automático, área de la lógica

computacional que actualmente cuenta con aplicaciones no triviales y que se origina en gran medida en la programación lógica.

Elementos de programación lógica y PROLOG

En este último capítulo el autor nos presenta la versión particular de la regla de resolución binaria, que constituye el motor de inferencia del lenguaje de programación PROLOG, además de algunas particularidades de este lenguaje de programación. Después de un oportuno comentario histórico y de una conveniente discusión de las generalidades de la programación lógica, en particular de la interpretación procedimental de la lógica clausular, con la que el autor aclara de manera precisa porqué considerar a un conjunto de cláusulas como un programa lógico no es un abuso, se presenta el principal refinamiento de la resolución, la llamada resolución lineal con función de selección en cláusulas definidas (resolución SLD) y se mencionan otros refinamientos como la resolución lineal o la resolución por cláusulas positivas, omitiéndose una estrategia trascendental para el razonamiento automatizado, como es la resolución semántica, en particular la llamada estrategia del conjunto de soporte. Enseguida se presentan algunos ejemplos de ejecución de programas y se discute la versión particular de la regla de resolución SLD para PROLOG, a saber, la que siempre resuelve la primera literal de la cláusula meta contra la primera cláusula del programa cuya cabeza unifique con dicha literal. Los siguientes apartados utilizan variaciones de un ejemplo clásico acerca de vínculos familiares para mostrar de manera clara y eficaz el algoritmo de resolución de PROLOG, el surgimiento de ramas infinitas de búsqueda y las bondades y defectos de de la exploración del árbol de resolución por profundidad y por amplitud, así como el proceso de búsqueda por retroceso.

De Ledesma termina su monografía exponiendo dos características no lógicas del lenguaje, los operadores aritmé-

ticos primitivos y el predicado de corte que permite al programador tener control directo sobre el programa. Acerca de este último la exposición se da, de manera muy acertada, en base a un ejemplo sencillo e incluye una explicación sobre el significado procedimental del operador de corte, situación que será ampliamente agradecida por los programadores de PROLOG que tengan este libro en sus manos.

Bibliografía

BRAMER, M. 2005. *Logic Programming with Prolog*. Springer Verlag.

HALPERN, J. Y. R., IMMERMAN, Harper N., KOLAITIS, P. G., VARDI, M. Y., VIANU, V. 2001. "On the Unusual Efectiveness of Logic in Computer Science". *Bulletin of Symbolic Logic*. **7**2: 213-236.

HUTH, M., RYAN. M. 2004. *Logic in Computer Science, Modelling and Reasoning about Systems.* Second Edition, Cambridge University Press.

LEITSCH, A. 1997. *The Resolution Calculus. Texts in Theoretical Computer Science*. Springer Verlag

LLOYD, J.W. 1987. *Foundations of Logic Programming*. Springer Verlag. Second, extended edition.

SHOENFIELD, J. R. 2001. *Mathematical Logic*. Association for Symbolic Logic.

SPERSCHNEIDER, V., ANTONIOU, G. 1991. *Logic: A Foundation for Computer Science*. Addison-Wesley.

STERLING, L., Shapiro, A. 1994. *The Art of Prolog*. Second edition, MIT Press.