ISSN: 0213-3563

ELIJA SU PROPIA LÓGICA

Choose your own Logic

Carlos ARECES
Inria Lorraine, Nancy, areces@loria.fr, web: www.loria.fr/~areces

BIBLID [(0213-356)8,2006,71-83]

Fecha de aceptación definitiva: 20 de abril de 2006

RESUMEN

En este artículo se sintetiza una visión moderna de las lógicas modales y temporales. En vez de dar una motivación histórica, el paper presenta estas lógicas en relación con ciertos fragmentos de la lógica de primer orden que poseen propiedades interesantes. Esta visión de la lógica es seductora porque nos permite diseñar lenguajes a medida, es decir, optimizados para una tarea específica.

Palabras clave: Lógicas modales, lógicas temporales, fragmentos de la lógica de primer orden, traducciones, el problema de clasificación.

ABSTRACT

This paper presents a modern vision of the field of modal and temporal logics. Instead of being historically motivated, the paper presents these logics in conexion with some fragments of first order logic with interesting properties. This approach is seductive because it allows us to view modal logics as languages designed «to size», i.e., to fit a given task.

Key words: Modal logics, temporal logics, fragments of first order logic, translations, classificatiom problem.

© Ediciones Universidad de Salamanca

1. La Bella y la Bestia

La lógica de primer orden es un lenguaje hermoso. Como lógica, es una gran candidata a ser la «Bella» de esta historia. Es un lenguaje elegante, simple de caracterizar y de buen comportamiento teórico. Pero cuando la miramos desde un punto de vista computacional –por ejemplo, si queremos usarla como lenguaje de especificación del conocimiento básico de un robot– entonces se transforma en la «Bestia». Ya que no existe un algoritmo que nos permita decidir si una fórmula de la lógica de primer orden es satisfacible (es decir, si la fórmula tiene al menos algún modelo, cualquiera que sea éste).

Aún el problema de saber si una fórmula de primer orden es satisfacible en un modelo dado es difícil. Es decir, aún si nos dicen «aquí está la sentencia φ, y aquí está el modelo $\mathbf{M}$ (finito), por favor dígame si φ es cierta en $\mathbf{M}$'» ese problema (que se llama *chequeo de modelos*) no tiene una solución eficiente: puede requerir espacio polinomial y tiempo exponencial (es un problema en PSpace). En una aplicación, un comportamiento de tiempo exponencial es usualmente inaceptable.

Para entender por qué miremos un ejemplo. Supongamos que la fórmula φ tiene 5 operadores, y que el modelo $\mathbf{M}$ no es demasiado grande, digamos 10 elementos. Entonces, dependiendo de la estructura exacta de φ y de $\mathbf{M}$' chequear que φ es cierta en $\mathbf{M}$ nos puede llevar 10^5 pasos. Veamos, si cada paso nos toma un segundo de computación entonces:

$$10^5 = 100{,}000 \text{ pasos o segundos} \sim 28 \text{ horas}$$

Bueno, es cierto que un segundo es mucho tiempo para una computadora, y que en ese tiempo es posible que realice miles de operaciones y no una sola operación como dijimos arriba. Consigamos entonces una supercomputadora que pueda realizar un millón de pasos por segundo. Quizás ahora no tendríamos problemas? No en el ejemplo que consideramos antes (que resolveríamos en sólo 0, 1 segundos), pero consideremos una fórmula de tamaño 10, y un modelo de 25 elementos. Entonces:

$$25^{10} = 95{,}367{,}431{,}640{,}625 \text{ pasos} \sim 95{,}367{,}431 \text{ segundos} \sim 26{,}490 \text{ horas}$$

Contemos como contemos y no importa cuan rápida sea nuestra computadora, problemas que requieran tiempo exponencial siempre tendrán instancias demasiado difíciles de resolver.

Pero las noticias no son tan malas como parecen. No todas las fórmulas de la lógica de primer orden son tan difíciles como los ejemplos que discutimos más arriba. Recordemos que decir que un determinado problema está en una clase de complejidad C (como PSpace o ExpTime) quiere decir que existe *alguna* instancia del problema que requiere tanto espacio o tiempo. Podría haber muchas otras instancias mucho más simples.

Y aquí es donde el tema se pone interesante. Seguramente en una aplicación dada no usaremos *todo* el poder expresivo de la lógica de primer orden. ¿Quizás es posible *elegir* fragmentos más simples y que de todas formas tengan la expresividad necesaria para nuestra aplicación?

© Ediciones Universidad de Salamanca

2. Frag men tos

Entonces, la idea es inspeccionar el conjunto de todas las fórmulas del lenguaje de primer orden y ver si podemos detectar subconjuntos «interesantes». Tenemos dos problemas, primero ¿cuál es una forma razonable de elegir fragmentos de primer orden? Y segundo ¿qué significa «interesantes»?

Empecemos por la segunda pregunta. Como primer requerimiento, el conjunto de fórmulas que elijamos tiene que ser suficientemente genérico. Podríamos pensar que un conjunto finito de fórmulas $\{\varphi_1,\ldots,\varphi_n\}$ es quizás adecuado para una determinada aplicación. Quizás pensamos que $\{\varphi_1,\ldots,\varphi_n\}$ son los axiomas de una teoría que estamos desarrollando, o las fórmulas que queremos que nuestro robot «conozca». Pero debemos considerar también como miembros de nuestro conjunto de fórmulas las preguntas que queremos que el robot pueda contestar, o los teoremas que queremos poder derivar a partir de los axiomas de nuestra teoría. Esto pronto haría que nuestro conjunto se vuelva infinito. Queremos entonces elegir conjuntos infinitos de fórmulas que de alguna forma tengan algo en común, y que sean a la vez útiles en aplicaciones y más simples de tratar.

Llegamos así a la primera pregunta. A priori, una forma natural de obtener fragmentos del lenguaje de primer orden sería imponer límites a los «recursos» que el lenguaje posee. Pero ¿cuáles podrían ser estos recursos? Por ejemplo, notemos que las variables de una fórmula de primer orden funcionan como células de memoria donde podemos almacenar información. Además, son justamente las variables sobre lo que actúan los operadores de cuantificación (cuando escribimos $\forall x{\cdot}\varphi(x)$ estamos diciendo, «pruebe con todo posible valor v de x y verifique que $\varphi(\mathrm{x}{:=}v)$ es cierta»). Podemos entonces considerar los fragmentos $\mathrm{LPO}^{\mathrm{k}}$ para $k \in$ IN de fórmulas de primer orden con a lo sumo un número dado de variables. Es decir, en cada LPO^{k} pueden aparecer a lo sumo las primeras k variables $x_1,\ldots,x_k$ disponibles en nuestro vocabulario de primer orden.

Ya LPO^{2} (i.e., las fórmulas de primer orden con sólo dos variables) es un fragmento interesante. Por ejemplo, pensemos cómo describiríamos un camino de longitud n en un grafo. Probablemente nuestro primer intento sería

$$\exists x_1\ldots\exists x_n{\cdot}(\mathit{Eje}(x_1,x_2) \wedge \ldots \wedge \mathit{Eje}(x_{\mathrm{n1}},x_n))$$

que dice, simplemente, que existen n nodos en el el grafo (no necesariamente diferentes) que están conectados por la relación *Eje*. Notar que usamos n variables diferentes para escribir esta fórmula.

Pero podemos escribir la misma propiedad usando solamente *dos* variables x_1 y x_2, si las reusamos acomodando adecuadamente los cuantificadores (i.e., si usamos nuestras células de memoria cuidadosamente):

$$\exists x_1{\cdot}\exists x_2{\cdot}(\mathit{Eje}(x_1,x_2) \wedge x_1{\cdot}(\mathit{Eje}(x_2,x_1) \wedge\ldots))$$

Para entender esta nueva forma de representar un camino de longitud n, podemos pensar que estamos avanzando paso a paso. En el primer paso nos movemos

© Ediciones Universidad de Salamanca

del punto del grafo que almacenamos en x_1 al punto que almacenamos en x_2. Ya en el segundo paso, podemos reusar nuestra célula de memoria x_1 y asignarle el valor de nuestro tercer punto en el camino. Esto es exactamente lo que estamos haciendo cuando escribimos $\exists x_1$ por segunda vez en la fórmula. (El ejemplo está tomado de [3], la referencia clásica en el tema de fragmentos decidibles de la lógica de primer orden).

LPO^2 es un fragmento decidible de la lógica de primer orden. Este resultado fue probado por primera vez por Dana Scott en [13] y extendido para el fragmento LPO^2 con igualdad por Mortimer en [9]. En una serie de artículos, Grädel *et al.* investigaron en detalle el tema de decidibilidad o indecidibilidad de distintos fragmentos cercanos a LPO^2 (ver por ejemplo, [5, 6]).

El problema con esta forma de obtener fragmentos de la lógica de primer orden es que LPO^3 es ya un lenguaje indecidible. Notemos que fórmulas muy útiles en aplicaciones (e.g., transitividad $\forall x{\cdot}\forall y{\cdot}\forall z{\cdot}[R(x, y) \wedge R(y, x){\rightarrow}R(x, z)]$ necesitan ya tres variables, y se puede demostrar que no son equivalentes a ninguna fórmula con menos variables.

Miremos entonces a otras alternativas. En realidad, la búsqueda por fragmentos simples pero interesantes de la lógica de primer orden (y principalmente fragmentos decidibles) es casi tan antigua como la lógica de primer orden misma. Y mucha de la literatura clásica del tema merece ser leída con atención, como por ejemplo los resultados de Ackerman [1]. En este artículo y en otros que siguieron, se definen fragmentos de la lógica de primer orden mediante lo que se llama *prefijos de cuantificación*: toda fórmula de primer orden puede escribirse, en forma equivalente, como una formula $Q{\cdot}\varphi$ *donde* Q es un bloque ininterrumpido de operadores de cuantificación y φ una fórmula sin cuantificadores. Por ejemplo la fórmula:

$$\forall x{\cdot}\forall y{\cdot}\forall z{\cdot}(R(x, y) \wedge R(y, x){\rightarrow} \exists x{\cdot}[R(y, x) \wedge R(z, x)]$$

puede reescribirse como:

$$\forall x{\cdot}\forall y{\cdot}\forall z{\cdot}\exists n{\cdot}[(R(x, y) \wedge R(x, z)]{\rightarrow} [R(y, n) \wedge R(z, n)]$$

(pero notemos que «pagamos un precio» por la reescritura, ya que debemos introducir una nueva variable n).

De esta forma, podemos separar las fórmulas de la lógica de predicados (módulo equivalencia) seleccionando diferentes prefijos de cuantificación. Volviendo a Ackerman's [1], allí se demuestra por ejemplo que decidir la satisfacibilidad de fórmulas en los fragmentos definidos por los prefijos $\forall^*\exists^*$ (esta notación indica cero o más cuantificadores universales seguidos de cero o más cuantificadores existenciales) y $\forall^*\exists\forall^*$ (cero o más cuantificadores universales, un cuantificador existencia, y cero o más universales) son decidibles (aunque de alta complejidad).

Algunos fragmentos que parecen a priori simples son en realidad muy expresivos. Por ejemplo Skolem demostró ya en 1920 que el fragmento $\forall^*\exists^*$ es lo que se llama una «clase de reducción para la lógica de primer orden», lo que significa

© Ediciones Universidad de Salamanca

que toda fórmula del lenguaje de primer orden tiene una fórmula equivalente en $\forall^*\exists^*$. *En otras palabras,* $\forall^*\exists^*$ es tan expresiva como todo el lenguaje (y, por lo tanto, indecidible). Gödel mostró en 1933 que bastaban tres cuantificadores universales, es decir $\forall^3\exists\forall^*$ ya captura la expresividad de todo el lenguaje.

El estudio de los distintos fragmentos definidos por prefijos de cuantificación ha sido exhaustivo. El libro *The Classical Decision Problem* de Börger, Grädel y Gurevich [3] que citamos más arriba contiene un mapeo completo de *todos* los fragmentos decidibles o indecidibles del lenguaje de primer orden que pueden definirse mediante prefijos de cuantificación. Además, también se conoce la complejidad de muchos de los fragmentos decidibles. Esta tarea (el mapeo completo de decidibilidad/indecidibilidad de todos los fragmentos del lenguaje de primer orden definidos por prefijos de cuantificación) es lo que se conoce en lógica clásica como el «Problema de Clasificación» y como acabamos de mencionar, fue completada gracias a resultados de algunas de la figuras más importantes de la lógica. En particular, uno de los resultados cruciales que permitió la solución total del problema es el llamado «Teorema de Clasificación» de Gurevich [7] que mostró que bastaba investigar la decidibilidad/indecidibilidad de sólo un número finito de fragmentos para, a partir de ellos, poder clasificar todos los demás.

El trabajo en el Problema de Clasificación es, quizás, uno de los resultados más relevantes de las últimas décadas en lógica clásica. Sin embargo, los fragmentos definidos vía prefijos de cuantificación tienen al menos dos desventajas. La primera desventaja (seria) es que todos ellos permiten sólo un número finito de alternación de cuantificación. Por definición, cada uno de estos fragmentos permite alternar los cuantificadores $\exists$ y $\forall$ (es decir, cambiar de $\exists$ a $\forall$ o viceversa) un número finito de veces. La fórmula:

$$\exists x_1 \cdot \forall x_2 \cdot \exists x_3 \cdot [\mathit{HijoDe}(x_1,x_2) \wedge \mathit{HijoDe}(x_2,x^3)]$$

tiene nivel de alternación 2 (cambiamos dos veces el tipo de cuantificación primer de $\exists$ a $\forall$ y luego otra vez a $\exists$) y dice que hay alguien cuyos hijos todos tienen al menos un hijo. Podemos continuar ese patrón y escribir:

$$\exists x_1 \cdot \forall x_2 \cdot \exists x_3 \cdot \forall x_4 \cdot \exists x_5 \left(\begin{array}{l} \mathit{HijoDe}(x_1,x_2) \wedge \\ \mathit{HijoDe}(x_2,x_3) \wedge \\ \mathit{HijoDe}(x_3,x_4) \wedge \\ \mathit{HijoDe}(x_4,x_5) \end{array} \right)$$

Que dice algo así como: hay alguien cuyos hijos todos tienen un hijo al que a su vez le pasa que todos sus hijos tienen al menos un hijo. Este tipo de frases son difíciles de leer (y usualmente nunca las usaríamos en el lenguaje de todos los días). Pero fórmulas de este tipo pueden surgir fácilmente en aplicaciones.

La segunda desventaja de los fragmentos definidos por prefijos de cuantificación parece, a primera vista, menos seria o quizás una contradicción en

© Ediciones Universidad de Salamanca

sí misma: el problema es que son *fragmentos*. Como niños malcriados, queremos toda la torta, y no sólo una porción de ella. Es mucho mejor una torta completa (aunque sea pequeña), que sólo una porción de una torta más grande.

En las siguientes páginas vamos a mostrar como podemos solucionar estos dos problemas, y obtener lenguajes que sean, al mismo tiempo, suficientemente expresivos, computacionalmente tratables, sin restricciones de alternación de cuantificadores y fragmentos pero no fragmentos. Estos lenguajes, son los llamados *lenguajes modales* y tienen también larga historia (el mismo Aristóteles se ocupó de ellos, junto con su famoso «Todos los hombres son mortales…»). Aquí, sin embargo, los presentaremos desde una perspectiva diferente. Para una presentación clásica, aconsejamos consultar el libro «Modal Logic» [2] de Blackburn, de Rijke y Venema, hoy por hoy la referencia más completa sobre el tema.

3. Lógicas para Todos los Gustos

Pero primero, y para mantener el suspenso, una digresión.

3.1. *Pensando Modalmente*

Los lógicos modales tenemos una manera peculiar de mirar el mundo.

Todo en el universo es un grafo.

Todo o casi todo. Consideremos, por ejemplo, los días de la semana. Un lógico modal los vería como:

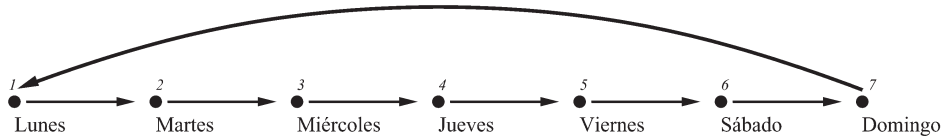


Figura 1: *Días de la Semana*.

Y en ese grafo nos pensamos en uno de sus puntos, por ejemplo, Jueves, y miramos hacia adelante («mañana va a ser Viernes») o hacia atrás («ayer fue Miércoles»), y a medida que el tiempo pasa avanzamos de punto en punto. En realidad, podemos argumentar que no somos sólo los lógicos modales los que vemos el tiempo de esa forma. Nuestro propio lenguaje tiene una perspectiva *interna* del tiempo, y al escuchar una frase como «Mañana hará exactamente un año desde que nos mudamos» construimos en nuestra mente una representación que tiene un punto para el *hoy,* de allí nos movemos hacia *mañana*, y desde allí retrocedemos un año hasta el momento de nuestra mudanza. (Esta perspectiva interna del tiempo, y el lenguaje adecuado para describirla, fue la contribución principal de Arthur Prior, unos de los padres de la lógica modal moderna [11, 12, 10]).

© Ediciones Universidad de Salamanca

Una fórmula modal se evalúa exactamente como acabamos de describir. Digamos que usamos el operador ⟨Mañana⟩ para movernos un día hacia adelante y el operador ⟨Ayer⟩ para movernos un día hacia atrás en nuestro modelo de la semana de la Figura 1. En este lenguaje modal escribiríamos «hoy es Jueves, mañana es Viernes y ayer fue Miércoles» como:

$$\text{Jueves} \wedge \langle \text{Mañana} \rangle \text{Viernes} \wedge \langle \text{Ayer} \rangle \text{Miércoles}$$

Notemos que ya en este lenguaje tan simple podemos expresar algunas ideas interesantes. Como el hecho siempre cierto de que si algo (llamémoslo p) pasa hoy, entonces será el caso que mañana pasará que ayer pasó p (siempre y cuando sepamos que va a haber un mañana).

$$(\text{p} \wedge \langle \text{Mañana} \rangle \text{T}) \rightarrow \langle \text{Mañana} \rangle \langle \text{Ayer} \rangle p$$

Pero para que las ideas queden claras, definamos nuestro primer lenguaje modal con un poco más de precisión. Vamos a obtener el lenguaje modal LM extendiendo el lenguaje Booleano, y seguiremos usando intuiciones temporales y nuestro operadores ⟨Mañana⟩ y ⟨Ayer⟩ (pero abreviaremos los nombres a ⟨M⟩ y ⟨A⟩ para simplificar un poco la notación). Definimos entonces, dado un conjunto de símbolos de proposición PROP que representarán hechos atómicos, el lenguaje LM de fórmulas modales como:

$$\text{LM: } = p \mid \neg \varphi \mid \varphi \wedge \psi \mid \varphi \langle \text{M} \rangle \varphi \mid \langle \text{A} \rangle \varphi$$

donde p es un elemento de PROP, y φ, ψ son fórmulas de LM (otros operadores Booleanos como $\vee$, $\rightarrow$ y $\leftrightarrow$ se introducen por definición en la forma usual).

Como dijimos, vamos a evaluar este lenguaje en grafos (más exactamente, grafos etiquetados) M = $\langle N, E, V \rangle$ donde N es el conjunto de nodos (no vacío), E es el conjunto de ejes, y V: $\rightarrow$ N 2^{PROP} es el conjunto de hechos atómicos que son ciertos en cada nodo, es decir para p $\in$ PROP, $p \in V(n)$ significa que p es cierta en el nodo n. Por ejemplo, representaríamos el grafo de la figura 1 como:

$$M = \langle N, E, V \rangle \text{ donde } \begin{cases} N = \{1, 2, 3, 4, 5, 6, 7\} \\ E = \{(1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 1)\} \\ V(1) = \{\text{Lunes}\} \ldots V(7) = \{\text{Domingo}\} \end{cases}$$

Podemos ahora ver que, dado el grafo M y un nodo n en el que nos pararemos, podemos evaluar si una fórmula φ de LM es cierta en dicho nodo usando el siguiente procedimiento:

1. Si $\varphi = p \in$ PROP entonces, ver si p $\in V(n)$
2. Si $\varphi = \neg\psi$ entonces ver si ψ no es cierta en n
3. Si $\varphi = \psi \wedge \theta$ entonces, ver si ψ y θ son ciertas en n
4. Si $\varphi = \langle \text{M} \rangle \psi$ entonces, ver si hay un nodo n' sucesor de n (i.e., $E[n, n']$) que haga cierta ψ).

© Ediciones Universidad de Salamanca

5. Si $\varphi = \langle A \rangle \psi$ entonces, ver si hay un nodo n' predecesor de n (i.e., $E(n', n)$ que haga cierta ψ)

Es un buen ejercicio para entender cómo funciona nuestro flamante lenguaje LM, intentar aplicar el procedimiento que acabamos de describir para evaluar una fórmula como:

$$\text{Lunes} \rightarrow \langle M \rangle \langle A \rangle \text{ Lunes}$$

en el nodo 1 del gráfico de la Figura 1. En realidad, podemos comprobar que la fórmula es cierta en todos los nodos de la figura, ya que en todos los demás nodos el antecedente Lunes falla.

Estamos listos ahora para dar un paso importante. Supongamos que queremos decir «me iré de vacaciones». Es decir, sé que en algún momento del futuro saldré de vacaciones, pero eso puede ser mañana o algún otro día, ¿cómo podemos expresar esto en nuestro lenguaje? Una posibilidad sería:

$$\langle \text{Mañana} \rangle \text{ de vacaciones} \vee$$
$$\langle \text{Mañana} \rangle \langle \text{Mañana} \rangle \text{ de vacaciones} \vee$$
$$\langle \text{Mañana} \rangle \langle \text{Mañana} \rangle \langle \text{Mañana} \rangle \text{ de vacaciones} \vee$$

* * *

pero las fórmulas infinitas son siempre difíciles de entender (y largas de escribir). Claramente, esta no parece la mejor solución. En cambio, extendamos nuestro lenguaje modal con un nuevo operador $\langle F \rangle$ y extendamos nuestro procedimiento de evaluación de fórmulas con la siguiente condición:

6. Si $\varphi = \langle F \rangle \psi$ entonces, ver si existen k nodos $n_1, \ldots, n_k$ tales que $E(n, n_1)$, $E(n_1, n_2)$, …, $E(n_{k-1}, n_k)$ y n_k hace cierta ψ.

Notar que la condición 6 no es más que una manera formal (y un poco engorrosa) de decir que para que $\langle F \rangle \psi$ y sea cierta en n debe haber algún nodo «en el futuro de n» que haga cierta ψ. (Este es en realidad el operador clásico de futuro de la lógica temporal de Prior, aunque usualmente se introduce de una forma diferente, restringiendo la clase de modelos a grafos donde la relación es transitiva. Las dos presentaciones no son exactamente equivalentes, pero son suficientemente similares para nuestros propósitos.)

La ventaja es que, una vez que introdujimos este nuevo operador, podemos fácilmente escribir «me iré de vacaciones» como:

$$\langle F \rangle \text{ de vacaciones}$$

© Ediciones Universidad de Salamanca

La idea principal detrás de esta forma de pensar lenguajes lógicos es la siguiente:

> Compilemos, una única vez, las nociones que nos interesan definiendo nuestros operadores. Y luego, podemos usarlos libremente, sin preocuparnos más de los detalles.

Veamos otro ejemplo para afirmar nuestras intuiciones. Sé entonces que me iré de vacaciones, pero para poder irme, tengo primero que terminar con todo mi trabajo acumulado. Es decir, «me iré de vacaciones, pero hasta que eso ocurra tendré muchísimo trabajo». Intentemos aplicar nuestra idea de «compilación de nociones interesantes» a este ejemplo. Para variar, usemos ahora la concepción mas clásica, pensemos que la relación E en nuestros grafos representa el orden entre instantes de tiempo: $E(i_1,i_2)$ significa que el instante i_1 es temporalmente anterior a i_2. Asumiremos entonces, por lo menos, que E es transitiva correspondiendo a nuestra noción natural del flujo del tiempo (i.e., si i_2 es un instante de tiempo posterior a i_1,y i_3 es posterior a i_2 entonces, claramente i_3 es posterior a i_1).

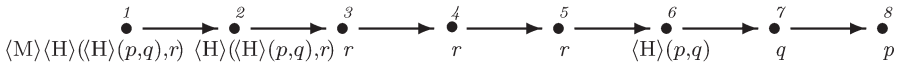
Definamos ahora un operador que capture las ideas detrás de nuestro último ejemplo. Claramente, necesitamos definir un operador binario «hastaque-pase-φ-pasará-φ.» Llamemos $\langle H\rangle$ a este operador, entonces:

7. Si $\varphi = \langle H\rangle(\psi, \theta)$ entonces, ver si existe un nodo n' sucesor de n [i.e., $E(n, n')$] que haga cierta ψ y para todo nodo intermedio] i.e., para todo nodo n" tal que $E(n, n")$y $E(n",n)$] tiene que ser cierta θ

Nuestros nuevos operadores son cada vez mas complejos, pero como dijimos antes, nos basta prestarles atención y verificar que capturan las ideas que nos interesan *una única vez*. Una vez definidos podemos usarlos tranquilamente para formar fórmulas simples que capturan nociones complejas. Podríamos escribir cosas como:

$$\langle M\rangle\langle H\rangle(\langle H\rangle(p, q),r)$$

Traducir este tipo de expresiones al castellano resulta en frases bastante complejas. Es mas fácil visualizar su significado en un grafo ejemplo que haga la fórmula cierta:



Pensemos que la relación entre los nodos es la clausura transitiva de los ejes mostrados en el grafo. Entonces $\langle H\rangle(p, q)$ es cierta en 6, porque en 8 es cierto p y en todos los puntos intermedios [solo 7 en este caso, es cierto q. $\langle H\rangle(\langle H\rangle(p, q),r)$ es cierta en 2, porque $\langle H\rangle(p, q)$ es cierta en 6 y en todos los puntos intermedios (3, 4 y 5] es cierta r. Y como $\langle H\rangle(\langle H\rangle(p, q),r)$ es cierta en 2, $\langle M\rangle\langle H\rangle(\langle H\rangle(p, q)r)$ es cierta en 1.

© Ediciones Universidad de Salamanca

Pero esta digresión ya se extendió quizás demasiado, e imagino que el suspenso creció suficientemente. Es buen momento para ver qué tienen que ver los lenguajes modales con los fragmentos del lenguaje de primer orden.

3.2. *Fragmentos Escondidos*

¿Donde están las variables? ¿Dónde están los cuantificadores? ¿Qué tiene que ver el lenguaje LM, por ejemplo, con nuestra querida lógica de primer orden? Veamos…

Primero, en la sección anterior dijimos que trabajaríamos con grafos etiquetados. Pero un grafo etiquetado no es otra cosa que un modelo particular de primer orden: el conjunto N de nodos es el dominio del modelo, la relación E es la interpretación de un símbolo de relación binario, y cada V (p_i) para un símbolo de proposición p_i puede tomarse como la interpretación de un símbolo de relación unario P_i. La relación a nivel semántica entre nuestra lógica LM y la lógica de primer orden es estrecha: *comparten el mismo tipo de modelos*.

La relación a nivel sintáctico es igualmente fuerte: *toda fórmula de* LM *tiene una fórmula de primer orden equivalente*. Definamos en detalle esta traducción que mapea fórmulas de LM en fórmulas de primer orden:

1.	$\mathrm{ST}_x(p_j)$	=	$P_j(x)$, $p_j \in$ PROP
2.	$\mathrm{ST}_x(\neg\psi)$	=	$\neg\mathrm{ST}_x(\psi)$
3.	$\mathrm{ST}_x(\psi \wedge \theta)$	=	$\mathrm{ST}_x(\psi) \wedge \mathrm{ST}_x(\theta)$
4.	$\mathrm{ST}_x(\langle \mathrm{M}\rangle\psi)$	=	$\exists y\cdot(E(x, y) \wedge \mathrm{ST}y(\psi))$
5.	$\mathrm{ST}_x(\langle \mathrm{A}\rangle\psi)$	=	$\exists y\cdot(E(y, x) \wedge \mathrm{ST}y(\psi))$

En las donde y en las cláusulas 4) y 5) es una variable nueva, no usada anteriormente. Quizás la traducción que acabamos de dar resulta conocida. Comparémosla con las condiciones 1) a 5) del procedimiento para evaluar fórmulas de LM que dimos anteriormente. En realidad, no hemos hecho otra cosa que escribir las condiciones de evaluación de las fórmulas de LM en lógica de primer orden. Miremos, como ejemplo, la traducción de la fórmula Lunes $\rightarrow \langle \mathrm{M}\rangle \langle \mathrm{A}\rangle$ Lunes. La traducción sería:

$$\mathrm{Lunes}(x) \rightarrow \exists y\cdot[E(x, y) \wedge \exists z.(E(z, y) \wedge \mathrm{Lunes}(z)]$$

Como habíamos mencionado antes, en un grafo donde sabemos que existe un «futuro» para x (i.e., un nodo y, tal que $E(x, y)$), la fórmula siempre será cierta.

Intentemos describir las características de esta traducción. Lo primero que veremos, es que las fórmulas de primer orden que corresponden a traducciones de fórmulas de LM no tienen restricción en alternación de cuantificadores. Usemos $[\mathrm{M}]\psi$ como abreviatura de $\neg\langle \mathrm{M}\rangle\neg\psi$ (de la misma forma que $\forall x\cdot\varphi$ puede verse como una abreviatura de $\neg\exists x\cdot\neg\varphi$). Entonces, la traducción de la fórmula:

$$\langle \mathrm{M}\rangle[\mathrm{M}]\langle \mathrm{M}\rangle p$$

© Ediciones Universidad de Salamanca

tendrá dos alternaciones: de existencial a universal, y otra vez a existencial (compruebe esto computando $ST_x(\langle M\rangle[M]\langle M\rangle p)$, no es difícil). Y nada nos impide continuar de la misma manera y escribir, por ejemplo, $\langle M\rangle[M]\langle M\rangle[M]p$ con tres alternaciones, etc. Intuitivamente (y esto puede demostrarse formalmente) el conjunto de fórmulas obtenidas a partir de LM vía ST_x no es un subconjunto de *ningún* fragmento definido mediante prefijos de cuantificación.

Por otro lado, si administramos nuestras variables un poco más cuidadosamente, podemos obtener una traducción usando solamente dos variables (i.e., dentro de LPO^2). El truco es el siguiente: usemos dos funciones mutuamente recursivas ST_x y ST_y.

$$\begin{array}{rcl|rcl}
ST_x(p_j) & = & P_j(x),\ p_j \in \text{PROP} & ST_y(p_j) & = & P_j(y),\ p_j \in \text{PROP} \\
ST_x(\neg\psi) & = & \neg ST_x(\psi) & ST_y(\neg\psi) & = & \neg ST_y(\psi) \\
ST_x(\psi \wedge \theta) & = & ST_x(\psi) \wedge ST_x(\theta) & ST_y(\psi \wedge \theta) & = & ST_y(\psi) \wedge ST_y(\theta) \\
ST_x(\langle M\rangle\psi) & = & \exists y\cdot(E(x, y) \wedge ST_y(\psi)) & ST_y(\langle M\rangle\psi) & = & \exists x\cdot(E(y, x) \wedge ST_y(\psi)) \\
ST_x(\langle A\rangle\psi) & = & \exists y\cdot(E(y, x) \wedge ST_y(\psi)) & STy(\langle A\rangle\psi) & = & \exists x\cdot(E(x, y) \wedge STx(\psi))
\end{array}$$

Es decir cuando estamos usando ST_x y necesitamos una variable para el operador modal usamos y y, viceversa, cuando estamos usando ST_y usamos x como nuestra nueva variable. (Notemos que acabamos de mostrar que LM es decidible puesto que podemos verla como un fragmento de LPO^2).

Pero intentemos ahora traducir nuestro operador $\langle H\rangle$ usando la cláusula 7) del procedimiento de evaluación que definimos en la sección anterior:

$$ST_x(\langle H\rangle(\psi, \theta)) = \exists y\cdot[E(x, y) \wedge STy(\psi) \wedge \forall z\cdot(E(x, z) \wedge E(z, y) \rightarrow STz(\theta)]$$

Como vemos, ahora necesitamos tres variables en nuestra traducción (es decir, la traducción ahora produce fórmulas de LPO^3). Como dijimos, LPO^3 es un fragmento indecidible de la lógica de primer orden, y sin embargo el fragmento que obtenemos como imagen de LM + $\langle H\rangle$ es decidible (El operador que llamamos $\langle H\rangle$ no es otro que el operador Until introducido por Kamp en [8], y es sabido que este lenguaje es decidible y de buen comportamiento computacional).

Finalmente, intentemos traducir $\langle F\rangle$, a partir de la condición 6) de la sección anterior. Notemos que en este caso llegamos a un fragmento de la lógica clásica aún mas complejo: necesitamos hacer referencia a la clausura transitiva de la relación E para poder expresar la condición «un sucesor n' a un número finito, pero no previamente especificado, de pasos de n». Usando E^+ para la clausura transitiva de E, escribiríamos:

$$ST_x(\langle F\rangle\psi) = \exists y\cdot[E^+(x, y) \wedge STy(\psi)]$$

En este caso entonces, las formulas que obtenemos traduciendo LM +$\langle F\rangle$ no son siquiera un fragmento de la lógica de primer orden (ya que la lógica de primer orden no es suficientemente expresiva como para expresar clausura transitiva). Estaríamos capturando un fragmento de la lógica de segundo orden, o al menos uno de la lógica de primer orden extendida con un operador de clausura transitiva.

© Ediciones Universidad de Salamanca

Y sin embargo, LM + ⟨F⟩ es un lenguaje decidible y otra vez, de buen comportamiento computacional.

4. Ingeniería Lógica

Desde un punto de vista moderno, las lógicas modales pueden verse como formas de capturar fragmentos de lenguajes clásicos (de primer o alto orden), pero utilizando un criterio ingenieril. En vez de definir fragmentos en forma categórica imponiendo, por ejemplo, restricciones en los recursos existentes (sean estos número de variables, patrones de cuantificación, o algún otro), encapsulamos en nuevos operadores modales las nociones que nos interesa capturar (como hicimos con los operadores ⟨M⟩, ⟨F⟩ y ⟨H⟩). De esta forma, definimos una especie de «lenguaje mínimo» para una aplicación dada, y en muchos casos, los lenguajes obtenidos de esa forma poseen propiedades interesantes [14, 4].

De la misma forma que un ingeniero decide en que lugares colocar los pilares para que un puente se sostenga y pueda soportar determinada carga, el ingeniero lógico define operadores que permitan expresar los conceptos interesantes en una determinada tarea. Operadores que den al lenguaje el poder expresivo requerido, ni más ni menos.

Aquellos que conozcan de antemano los lenguajes modales, quizás se vean sorprendidos por esta presentación del área. Clásicamente, los lenguajes modales son pensados como lenguajes «intencionales» cuyo objetivo es capturar nociones como *necesidad*, *obligación,* etc. Pero las dos presentaciones son totalmente compatibles. Podemos tomar las palabras (ligeramente editadas) de Ramón de Campoamor[1] y decir:

> En este mundo seductor
> nada es verdad ni es mentira;
> todo es según el color
> del cristal con que se mira.

Porque en realidad es una visión seductora de la lógica. Donde cada uno puede diseñar el lenguaje que mejor le quede y que más contento le ponga. Anímese, elija su propia lógica.

1. Los versos originales son: *En este mundo traidor, nada es verdad ni es mentira; todo es según el color del cristal con el que se mira.*

© Ediciones Universidad de Salamanca

BIBLIOGRAFÍA

[1] ACKERMANN, W., *Solvable Cases of the Decision Problem*, Amsterdam, North-Holland, 1954.
[2] BLACKBURN, P.; DE RIJKE, M. and VENEMA, Y., *Modal Logic*, Cambridge, University Press, 2001.
[3] BÖRGER, E.; GRÄDEL, E. and GUREVICH, Y., *The Classical Decision Problem*, Springer Verlag, 1997. Los versos originales son: *En este mundo traidor, nada es verdad ni es mentira; todo es según el color del cristal con que se mira*.
[4] GRÄDEL. E., «Why are modal logics so robustly decidable?», *Bulletin of the European Association for Theoretical Computer Science, EATCS*, 68: 90-103, 1999.
[5] GRÄDEL, E.; OTTO, M. y ROSEN, E., «Two-variable logic with counting is decidable», in *Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science*, Warsaw, Poland, 1997, pp. 306-317, IEEE Computer Society Press.
[6] GRÄDEL, E.; OTTO, M. y ROSEN, E., «Undecidability results on twovariable logics», *Archive for Mathematical Logic*, 38 (4-5): 313-354, 1999. Logic Colloquium'95 (Haifa).
[7] GUREVICH, Y., «The decision problem for logic of predicates and operations», *Algebra and Logic*, 8 (160-3174), 1969.
[8] KAMP, J., *Tense Logic and the Theory of Linear Order*, PhD thesis, University of California, Los Angeles, 1968.
[9] MORTIMER, M., «On languages with two variables», *Z. Math. Logik Grundlagen Math*, 21: 135–140, 1975.
[10] ØHRSTRØM P. y HASLE, P., «A. N. Prior's rediscovery of tense logic», *Erkenntnis*, 39: 23-50, 1993.
[11] PRIOR, A., *Past, Present and Future*, Oxford, Clarendon Press, 1967.
[12] PRIOR, A., *Papers on Time and Tense*, Oxford, University Press, 1968.
[13] SCOTT, D., «A decision method for validity of sentences in two variables», *Journal of Symbolic Logic*, 27: 377-377, 1962.
[14] VARDI, M., «Why is modal logic so robustly decidable?», *Descriptive complexity and finite models (Princeton, NJ, 1996)*, pp. 149-183. Amer. Math. Soc., Providence, RI, 1997.

© Ediciones Universidad de Salamanca