
**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**

**POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN**

# RESULTADOS EN LÓGICA DE JUSTIFICACIÓN Y UNA IMPLEMENTACIÓN DEL TEOREMA DE ELUCIDACIÓN

**T E S I S**

**QUE PARA OPTAR EL GRADO DE:**

**MAESTRO EN CIENCIAS (COMPUTACIÓN)**

**P R E S E N T A :**

**JESÚS MAURICIO ANDRADE GUZMÁN**

**DIRECTOR DE TESIS:**
**DR. FRANCISCO HERNÁNDEZ QUIROZ**
**FACULTAD DE CIENCIAS - UNAM**
**MÉXICO, D.F., ENERO DE 2016**

Para Letycia

# Agradecimientos

Agradezco a la Universidad Nacional Autónoma de México (UNAM), al Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), al Consejo Nacional de Ciencia y Tecnología (CONACYT). A mi tutor, el Dr. Francisco Hernández Quiroz, por su paciencia y guía durante el proyecto; y a mis sinodales, por sus observaciones y comentarios.

## Índice general

# 1. Introducción

## 1.1. Antecedentes y motivación

La lógica de justificación es un aparato formal para operar sobre razones para una afirmación. Está inspirada por la lógica modal pero sustituyendo el carácter implícito de las modalidades por términos explícitos que funcionan como justificaciones. Para contextualizar lo que la lógica de justificación pretende cambiar, es necesario al menos retomar una parte del origen de la lógica modal.

La lógica modal contemporánea surge hace ya cerca de 100 años, con el trabajo de C. I. Lewis (Lewis & von Leibniz, 1918). En términos modernos, Lewis busca sustituir la idea de implicación material ($p \rightarrow q$), equivalente a ($\neg\, p \vee q$) por otra noción, en parte por las «paradojas de la implicación material» que no parecen dar cuenta de la relación entre el antecedente y el consecuente. Por tal motivo, propone la *implicación estricta*, que podemos representar como $\Box(p \rightarrow q)$, que captura la idea de necesidad y es verdadera solo si no es posible que el antecedente sea verdadero y el consecuente sea falso al mismo tiempo. Esta idea, en términos más actuales, significa que para que la implicación estricta sea verdadera, debe ser el caso en todas las posibles situaciones imaginables, estados o mundos posibles.

Sería difícil dar una definición completa de lo que hoy en día se conoce como lógica modal, porque es la combinación de avances en diversas áreas. El estudio de disposiciones como posibilidad y necesidad se puede rastrear incluso a la Antigua Grecia. La caracterización formal de Lewis (en **S1**, **S2**, **S3**, etc.) de estos aparatos

lógicos a través de grupos de axiomas que describen propiedades del sistema lógico, sobrevive a nuestros días. Actualmente, esta caracterización se ha visto influenciada con los avances en la teoría de modelos de Tarski, la semántica de mundos posibles de Kripke, y muchos otros que han contribuido en esta disciplina.

Las modalidades que hoy se estudian son tan diferentes entre sí que a veces se tratan como sistemas lógicos separados, *p. ej.* lógica temporal, lógica epistémica, lógica de demostraciones, etc. Este tipo de lógicas han desembocado en aplicaciones tecnológicas en el ámbito de la computación como inteligencia artificial y lenguajes de programación; de las comunicaciones en el diseño de protocolos para la transmisión de datos; en matemáticas como el punto de partida para hablar de demostraciones matemáticas como objetos de estudio; entre muchas otras aplicaciones. Cada una se estudia con un propósito en mente muy específico, siendo de mención especial la lógica epistémica, que se refiere a la formalización de afirmaciones sobre lo que conocemos.

La lógica epistémica es el estudio, a través de un lenguaje formal, de afirmaciones de conocimiento. En 1962, Hintikka (Hintikka, 1962) le dio una formalización al conocimiento que se ha hecho tradicional en la disciplina. Sin embargo, incluso con estos avances, al hablar de conocimiento, los problemas presentes en la epistemología filosófica son persistentes en las formalizaciones, incluso con la definición de conocimiento como una «creencia verdadera justificada», considerada la tradicional y más aceptada manera de referirse a lo que podemos conocer. En particular, Edmund Gettier (Gettier, 1963) en un pequeño artículo de 3 páginas, desafía esta definición. Gettier pone en evidencia los problemas en los criterios de jus-

tificación aceptados para afirmaciones de conocimiento. Desde entonces, las formalizaciones del conocimiento y las teorías filosóficas de justificación han cambiado mucho, pero aún se debate sobre el papel de la justificación y sobre si el conocimiento está correctamente caracterizado por la definición tripartita de Platón.

En este trabajo se presentará una familia de lógicas que hacen explícita la idea de justificación para afirmaciones de conocimiento. La lógica de justificación se puede ver como un tipo de lógica modal, pero las modalidades son sustituidas por elementos de un conjunto que representan la razón de las afirmaciones que se hagan. Este ajuste a las modalidades puede resultar simple, pero da a la lógica de justificación el poder de hablar de afirmaciones complejas que se habían resistido a una formalización adecuada. En particular, la lógica de justificación puede formalizar la lógica intuicionista (S. N. Artemov, 1998) e incluso dar una respuesta convincente al problema de la omnisciencia lógica como se verá en el capítulo § 4.

Es importante mencionar que la lógica de justificación (o cualquier formalización del conocimiento) solo es un instrumento para clarificar las relaciones entre los elementos de los que se habla. Hago especial énfasis en este punto porque la propuesta que presento en este trabajo no pretende ser normativa sobre la idea de conocimiento y no debe interpretarse como una postura filosófica sobre *lo que es* el conocimiento, sino *lo que se puede hacer* con él. Shapiro en (Shapiro, 1998) hace una distinción (respecto a la noción de consecuencia lógica), entre la lógica como *descripción* y la lógica como *modelo*. La lógica como *descripción* parte de que los enunciados de la lógica corresponden con la estructura del mundo, es decir, una visión monista de la consecuencia lógica (una sola «correcta» definición de lo que es la

consecuencia lógica); la lógica como *modelo,* por otro lado, parte de que la lógica en realidad modela los fenómenos pero no necesariamente en una correspondencia uno a uno con lo que se estudia, esta visión de la lógica acepta diferentes definiciones de consecuencia lógica, de acuerdo, por ejemplo, al éxito de la formalización elegida.

Resulta más o menos claro que la teoría de modelos y la manera de abordar la lógica modal actualmente es pluralista respecto a la noción de consecuencia lógica. Para cada aparato distinto se define la noción particular de consecuencia lógica que se utilizará. Por ejemplo, la consecuencia lógica para una lógica modal **K** es distinta a una lógica **S4**, debido a la diferencia en el conjunto de axiomas que acepta cada una.

La lógica vista de esta manera se puede entender como un diagrama o un mapa, un modelo del pensamiento, no como fundamento del conocimiento ni de sus relaciones para el entendimiento. La idea clásica de lógica se ha usado de esta manera al plantear un argumento en forma de premisas y conclusiones, sin restringir a un conjunto de reglas específico y relaciones entre elementos de un tipo bien definido. Así el lenguaje formal de la lógica de justificaciones no debe verse como la única manera de hablar de justificaciones de conocimiento, pero sí como una herramienta útil para hablar de manera explícita de justificaciones para enunciados que representen conocimiento. Las justificaciones son solo entidades abstractas como los números o las variables proposicionales, y podemos hablar de su estructura y relaciones entre estas entidades. La lógica de justificación no analiza directamente qué significa «$s{:}\,P$», es decir, «$s$ justifica $P$», más allá de la relación

entre el término de justificación $s$ y la proposición $P$, del mismo modo que en una *disyunción* en lógica proposicional no se analiza qué significa «$p \vee q$», sino solo la relación que existe entre estas fórmulas y su tabla de verdad, que damos por correcta de forma axiomática.

La lógica de justificación se deriva de la lógica de demostraciones **LP** (*cf. Logic of Proofs*), que corresponde a una versión explícita de **S4,** por los axiomas que acepta. Uno de los resultados más importantes en lógica de justificación es que una fórmula válida en el lenguaje de **LP** es también válida en **S4** a través de lo que se conoce como *proyección de olvido* (*cf. forgetful projection*), en donde, a partir fórmulas en **LP** con afirmaciones explícitas, se puede encontrar una expresión modal implícita en **S4** equivalente. El paso inverso también es posible, es decir, se puede encontrar una expresión explícita de fórmulas en **S4** y un polinomio de justificación en **LP**, a este paso se le conoce como *teorema de elucidación* (*cf. Realization Theorem*), con la intuición de que, a partir de una afirmación de conocimiento en **S4** de carácter implícito con la forma $\Box P$, se llega a una fórmula explícita con la forma $t{:}\, P$, que se lee «$t$ es una razón para $P$».

Para redondear estas ideas y enmarcarlas en un contexto computacional, se desarrolló una implementación del cálculo de inferencia de la lógica de justificación, que se puede interpretar como una implementación del teorema de elucidación que, a partir de una fórmula modal válida, es posible calcular la forma de la justificación en términos explícitos. En este caso, la forma del polinomio de justificación corresponde con la demostración expresada en el lenguaje de programación que utiliza el asistente COQ.

Melvin Fitting dio un paso importante en este sentido en (Fitting, 2013), con una implementación en PROLOG que calcula el polinomio de justificaciones a partir de un teorema de **S4**. La implementación que se propone en este trabajo se hará utilizando el asistente de demostraciones COQ (https://coq.inria.fr), por la importancia que tiene actualmente en la formalización de teorías matemáticas en un entorno computacional interactivo, y por su amplia difusión en implementaciones de este tipo, que involucran tanto sistemas deductivos, como semánticas de lenguajes de programación.

## 1.2. Objetivo

Los objetivos puntuales para este trabajo son:

1. *Presentar los avances recientes en lógica de justificación para dar a conocer su potencial y el impacto que ha tenido.*

2. *Desarrollar una implementación computacional del cálculo inferencial de lógica de justificación, así como una implementación del teorema de elucidación.*

Para el primero, será necesario analizar lo que hace distinta a lógica de justificación de otras maneras de hablar del conocimiento. En un sentido, este es el resultado más fuerte del trabajo, porque la lógica de justificación se puede ver como una reformulación de la lógica modal, una reforma a ideas que ya están bien establecidas en la filosofía, las matemáticas y la computación. La intención es dar a conocer al investigador un aparato más potente con el que se puedan expresar nociones de conocimiento más detalladas, atendiendo a un elemento que represente la evidencia de las afirmaciones. Esto tiene muchas aplicaciones potenciales; pensemos en

un lenguaje de programación como PROLOG, pero en el cual cada *hecho* o *regla* que se codifique tenga también un elemento que represente una razón para creer en ese *hecho* o *regla*[1]. Por supuesto, esta idea no es nueva; por ejemplo, podemos encontrar en las implementaciones computacionales, como la del mismo COQ, consideraciones similares, como que el *tipo* de una afirmación es su *demostración*[2]. Son las ventajas computacionales que ofrece una definición recursiva y decidible de la lógica de justificación lo que hace atractiva a esta nueva formulación.

En Filosofía, y en particular en teoría del conocimiento, utilizar esta lógica permitiría hablar y caracterizar nociones de conocimiento de manera más detallada, con la confianza de que se está trabajando con un aparato lógico bien establecido, tal como ya se hace hoy en día con la lógica modal epistémica. Esta especie de «calculadora epistémica» permitiría hacer precisas las discusiones sobre algunos aspectos que se han resistido a formalización. Un lenguaje formal que permite hablar de justificaciones haría posible establecer la diferencia entre distintos tipos de evidencia para una afirmación, y expresar explícitamente la evidencia de afirmaciones de conocimiento (lo que se hace con la lógica epistémica de manera implícita con la relación de accesibilidad). Por mucho tiempo se ha debatido sobre si la lógica epistémica solo alcanza para representar nociones doxásticas y no de conocimiento,

[1] Un programa en PROLOG consiste de fórmulas lógicas codificadas como hechos y reglas para definir predicados, que son los que determinan la acción del intérprete.

[2] Esto fundamentado en el isomorfismo Curry-Howard (Sørensen & Urzyczyn, 1998) implementado con la teoría de tipos con la que trabaja COQ, el cálculo de construcciones inductivas.

aquí la lógica de justificación también puede aportar nuevos elementos para alimentar el debate, ofrecer nuevos medios de diagnóstico, o quizás incluso dar una respuesta.

El segundo objetivo, la implementación del cálculo deductivo de la lógica de justificación, permitirá materializar el poder de esta lógica. Se podrán visualizar las propiedades y estructuras de las que habla la lógica de justificación. Algunas de las demostraciones de las que se hablarán en el cuerpo del texto se podrán verificar con esta implementación. Se trata, principalmente, de un modo de razonamiento de deducción natural pero enriquecido con los términos de justificación.

El teorema de elucidación recibirá un tratamiento especial por tratarse de uno de los resultados más importantes de este tema. El resultado de la elucidación de una fórmula modal implícita $\Box F$, es su forma explícita $t{:}F$, con un polinomio de justificación $t$ que funciona como una razón para la afirmación $F$. El polinomio $t$ contiene la información necesaria para reconstruir la demostración de $F$ en el lenguaje de la lógica de justificación. Esto implica que un polinomio de justificación puede ser utilizado para generar «copias» de $F$, es decir, el agente «aprende» a generar los pasos de derivación que conducen a aceptar a $F$ como una afirmación de conocimiento, una característica que sin duda podría ser de interés en Inteligencia Artificial.

## 1.3. Resumen de capítulos

En el segundo capítulo se describirá el lenguaje de la lógica de justificación, la semántica formal, su axiomatización y ejemplos de derivación. Se hará un análisis

de la interpretación de las fórmulas justificadas y su implicación al hablar de conocimiento.

En el tercer capítulo se profundizará en las diferencias con la lógica modal epistémica, poniendo énfasis en los casos problemáticos al hablar de conocimiento, en particular los casos que corresponden con lo que inicialmente señala Gettier. En este capítulo también se estudiarán las herramientas formales para relacionar fórmulas en lógica epistémica y fórmulas en lógica de justificación, es decir, la *proyección de olvido* y los *teoremas de elucidación*.

En el cuarto capítulo se abundará en el tratamiento y respuesta de la lógica de justificación a la omnisciencia lógica. En este apartado se analizarán las cuestiones de complejidad computacional que permiten caracterizar a un sistema lógico, en cuanto a si es o no omnisciente.

En el quinto capítulo se integrará la información expuesta hasta el momento para hablar de la implementación computacional que se desarrolló. En este apartado se hará una breve reseña de la implementación de Melvin Fitting y se contrastará con la propuesta de este trabajo.

# 2. Lógica de justificación

## 2.1. Sintaxis

### Nomenclatura

Las *variables proposicionales* del lenguaje pertenecen a un conjunto infinito (numerable), $PROP = \{p_0, p_1, \ldots, p_n, \ldots\}$, pero se utilizarán letras sin subíndice para mejorar la lectura, i.e. $p$ en lugar de $p_0$, $q$ en lugar de $p_1$, etc. La notación con subíndices será útil al hablar de la implementación computacional[3] y cuando se hable en términos de complejidad. Cuando contribuya a mejorar la lectura y sea claro en el contexto, también se utilizarán letras mayúsculas para hablar de fórmulas del lenguaje, por ejemplo $F$ o $P$, para referirse a una fórmula cualquiera.

Los símbolos para las *conectivas proposicionales* de disyunción, conjunción, implicación y negación serán $\vee$, $\wedge$, $\rightarrow$ y $\neg$, respectivamente

Para hablar de *esquemas de fórmulas* se utilizarán letras griegas minúsculas, $\varphi, \psi, \theta, \ldots$ Cada subexpresión de un esquema de fórmula puede ser sustituido por cualquier fórmula bien formada en el lenguaje del que se esté hablando.

[3] En COQ, por ejemplo, la variable proposicional $p_0$ se expresará como $(p\ 0)$, es decir, el predicado $p$, aplicado al número natural 0.

Una *lógica* $L = \{\mathcal{L}, \models, \mathcal{CS}\}$, estará definida por un lenguaje formal $\mathcal{L}$; su definición de consecuencia lógica, $\models$; y un conjunto de especificación constante $\mathcal{CS}$[4], que contiene los axiomas justificados que se aceptarán.

Se utilizarán diferentes símbolos para hablar de *modalidades* a lo largo del texto. El símbolo tradicional $\Box$ (que se puede leer como «caja») expresa la modalidad de necesidad, y su modalidad dual de posibilidad $\Diamond$ (que puede leerse como «diamante») se considerará como una abreviatura de $\neg\Box\neg$. También será posible utilizar letras mayúsculas para expresar modalidades en diferentes contextos, p. ej. $K$, $B$, $K_i$, que se leerá como «sabe que», «cree que» y «el agente $i$ sabe que», respectivamente.

**Lenguaje de la lógica de justificación**

El lenguaje de la lógica modal formaliza el conocimiento como una modalidad, es decir, un agente epistémico «sabe $P$» (o sabe que $P$), se representa con la modalidad de necesidad, $\Box P$ o $KP$ en un contexto epistémico, es decir, $P$ es el caso en todos los mundos accesibles desde el punto de vista del agente. Por otro lado, $\Diamond P$ o $\neg K \neg P$, expresa la idea de posibilidad, es decir, existe al menos un estado accesible donde $P$ es el caso. Esta es la convención de formalización que hace posible expresar afirmaciones complejas de conocimiento, por ejemplo, que un agente «sabe que

[4] Más adelante, en este capítulo, se definirá con precisión a este conjunto.

sabe *P*», se representaría: *KKP*. Para un resumen de fórmulas y propiedades del lenguaje de la lógica modal, consulte el apéndice § A.

Una propiedad particular de la lógica epistémica es que acepta el axioma

$$\mathbf{T}\text{: } KP \rightarrow P$$

es decir, si se conoce *P*, debe ser verdadero. Si a la afirmación *P* se le considera una creencia en este contexto, estamos ante el conocimiento como una creencia que debe ser verdadera. Sin embargo, esta idea de conocimiento no considera una noción explícita de justificación.

La importancia de tomar en cuenta la justificación de una afirmación epistémica ha sido atendida por la lógica de justificación, propuesta por Sergei Artemov (S. Artemov, 2008). En ella se da un carácter explícito a los términos de justificación que permiten razonar sobre evidencia para las creencias. Las lógicas de justificación son similares a las lógicas modales, pero los operadores modales son reemplazados por elementos de un conjunto de justificaciones, que se interpretan como razones para las afirmaciones. La primera lógica de justificación fue propuesta por Sergei Artemov en (S. N. Artemov, 1998), conocida como **LP**, de lógica de demostraciones (*cf. Logic of Proofs*). Usando **LP**, Artemov dio una semántica formal a la lógica intuicionista siguiendo las ideas de Gödel y Kolmogorov. La lógica de justificación es una familia de lógicas que comparten la característica de hacer explícitas las justificaciones de las afirmaciones que se hacen.

La lógica de justificación es un lenguaje basado en lógica modal que permite razonar sobre la justificación del conocimiento. En este lenguaje, una fórmula $P$ constituye conocimiento si tiene una justificación $t$ que funciona como una razón para creer $P$. En símbolos, «$t: P$» se lee «$t$ es una justificación de $P$». Que la justificación $t$ sea una razón para creer $P$, en esta lógica significa que $t$ se construyó utilizando las reglas del lenguaje y está asociada con $P$.

Este tipo de razonamiento atiende la tercera característica del conocimiento de Platón en un lenguaje formal, respecto a que se razona con una *creencia verdadera justificada*.

Formalmente, el lenguaje de la lógica de justificación está formado por:

1. El lenguaje de la lógica proposicional;
2. términos atómicos de justificación, que pueden ser variables $x_0, x_1, \ldots$ o constantes $a_0, a_1, \ldots$ ;
3. símbolos de función para términos de justificación: unario (!), y binarios (·), (+);
4. el operador (:) funciona para unir términos de justificación con fórmulas del lenguaje, «*término*: *fórmula*»

2.1.1. **Definición**. Los *términos de justificación t*, están dados por la gramática:

$$t := x_i \mid a_i \mid !t \mid t \cdot t \mid t + t \text{ , con } i \in \mathbb{N}$$

Se entenderá como *polinomio de justificación* a un término construido con esta gramática.

**Equivalencia entre términos de justificación**

La relación de igualdad '=' entre términos de justificación, se considerará como primitiva del lenguaje. Es decir, cuando se establezca la igualdad entre dos términos de justificación $u$ y $v$, tal que $u = v$, significa que ambos elementos se refieren al mismo término. En particular, la relación de igualdad $=$ tiene las siguientes propiedades (Enderton, 2001, p. 112):

- **Reflexividad**. Para todo término de justificación $x$, $x = x$.
- **Sustitución en fórmulas**. Para todo $x, y$, $x = y \rightarrow (\varphi \rightarrow \varphi')$, donde $\varphi'$ se obtiene de sustituir cero o más veces a $x$ por $y$.

2.1.2. **Definición**. Una *fórmula* $\varphi$ en lógica de justificación está dada por la siguiente gramática, donde $p_i$ es un átomo proposicional $(p_0, p_1, \ldots)$, $\top$ es la constante verdadero[5] y $t$ un término de justificación:

$$\varphi ::= p_i \mid \top \mid \neg\varphi \mid \varphi \rightarrow \varphi \mid t{:}\varphi, \text{con } i \in \mathbb{N}$$

[5] La razón de utilizar la constante verdadero es contar con un caso base en la implementación computacional en Coq.

Los axiomas de la lógica de justificación básica **LP**, representan las relaciones entre términos de justificación y fórmulas.

## 2.2. Axiomatización

### Axioma de Aplicación

La operación aplicación ($\cdot$) entre dos justificaciones $s$ y $t$, produce una justificación $(s \cdot t)$[6], tal que:

$$Si\ s{:}\ (P \to Q)\ y\ t{:}\ P,\ entonces\ (s \cdot t){:}\ Q$$

es decir,

$$s{:}\ (P \to Q) \to t{:}\ P \to (s \cdot t){:}\ Q$$

Una especie de *Modus Ponens* para hablar de afirmaciones justificadas, que corresponde al axioma **K** de la lógica epistémica convencional:

$$K(P \to Q) \to KP \to KQ$$

[6] En lógica modal se suele utilizar paréntesis cuadrados para agrupar modalidades complejas, por ejemplo, en lógica dinámica se utiliza $[\varphi]F$, para expresar que se debe cumplir la fórmula $\varphi$, «antes» que la fórmula $F$. Para el caso de polinomios de justificación, se utilizarán paréntesis redondos, esto porque un polinomio de justificación puede estar formado por una combinación de operaciones entre términos. El carácter dos puntos (:) será la manera de separar el polinomio de justificación de la fórmula. La razón de esto es que, siguiendo la idea tradicional de lógica modal, se tendría que utilizar paréntesis cuadrados incluso para términos simples, como $[t]{:}\ F$, en lugar de $t{:}\ F$, siendo la segunda la notación preferida por la literatura y es de más fácil lectura.

El axioma **K** ha sido identificado como el elemento clave en el problema de la omnisciencia lógica, y tiene que ver con la cerradura bajo implicación lógica y que el razonamiento sea implícito. La lógica de justificación resuelve este problema al poder operar explícitamente sobre lo que considera evidencia para las creencias. Este tratamiento granular permite razonar sobre la longitud de la justificación en una fórmula epistémica. La reformulación de este axioma y su versión explícita serán piezas clave para revelar el poder y las ventajas de la lógica de justificación en un contexto epistémico.

**Axiomas de Suma**

El nombre Suma viene de la idea de concatenar (añadir) dos demostraciones a una afirmación en teoría de demostraciones. A modo de analogía, se puede pensar en un tomo de enciclopedia que ofrezca razones para un enunciado, y ese tomo apilado con otro (que puede ser de la misma enciclopedia) siguen conteniendo la razón para el enunciado.

*Si* $s{:}\,P$*, con cualquier evidencia adicional* $t$*, la evidencia combinada* $(s + t)$ *sigue siendo una justificación de* $P$*.*

La operación suma (+), toma las justificaciones $s$ y $t$, y produce $(s + t)$, que es una razón para todo lo que justifique $s$ o $t$. Lo que se traduce en dos axiomas que corresponden a cada uno de los casos:

$$s{:}\,P \rightarrow (s + t){:}\,P$$

$$t{:}\,P \rightarrow (s + t){:}\,P$$

Una consecuencia de este axioma es la propiedad de monotonía para **LP**. Es decir, si $s{:}\,P$ es el caso, agregando otra razón ($t$), no cambia el valor de verdad de la afirmación.

**Axioma de Verificación (cf. proof checker)**

Corresponde a la afirmación, si el agente «sabe $P$», entonces «sabe que sabe $P$», es decir, la introspección positiva $KP \rightarrow KKP$, pero en versión explícita.

Representa la situación en la que un agente epistémico acepta $t$ como evidencia de $P$, con lo que se tiene también evidencia para $t{:}\,P$, donde $!\,t$ representa esa evidencia.

$$t{:}\,P \rightarrow !\,t{:}\,(t{:}\,P)$$

El axioma de verificación no es necesario para una lógica de justificación básica, porque se puede sustituir por una instancia del principio de internalización, que se analizará más adelante.

**Conjunto de especificación constante,** $CS$

Otra característica fundamental de la lógica de justificación es poder formalizar la idea de conjuntos de fórmulas que constituyan axiomas para un sistema lógico epistémico, con lo que le da un carácter explícito al conocimiento; es decir, es posible cuantificar el número de axiomas para un sistema lógico dado, y por lo tanto, también cuantificar el número de pasos de una derivación dentro del marco lógico epistémico que se esté estudiando. A partir de lo anterior, es posible hablar de

eficiencia en afirmaciones epistémicas, y cuantificar los recursos necesarios para hacer afirmaciones en sistemas lógicos epistémicos, es decir, la complejidad de adquirir conocimiento.

Aquí la idea intuitiva a la que se refiere la complejidad de adquirir conocimiento es que los agentes deben hacer algo adicional para adquirir conocimiento, que se puede interpretar como esfuerzo, memoria, tiempo, etc. Con esto se elimina el problema de la omnisciencia lógica al permitir razonar sobre justificaciones específicas para las afirmaciones epistémicas. También permite al agente cognitivo decidir sobre si la justificación es adecuada aunque la afirmación epistémica se cumpla en todos los mundos posibles, lo que intuitivamente se puede interpretar como que un agente puede descartar una afirmación de conocimiento porque la justificación no es la adecuada a pesar de ser consistente con sus creencias.

El conjunto de especificación constante $\mathcal{CS}$, está dado por:

$$\mathcal{CS} = \{c_0 : A_0, c_1 : A_1, \dots\}$$

donde cada $c_i : A_i$, es una fórmula que se considera verdadera, y cada $c_i$ es un término de justificación que no requiere de validación adicional. Es decir, para todo mundo $w \in \mathcal{W}$, se postula verdadero $\mathcal{M}, w \models c_i : A_i$, en la siguiente sección § (2.3) se definirá la semántica de mundos posibles.

De esta manera, se puede caracterizar una lógica de justificación de acuerdo con un conjunto de especificación constante. Un conjunto de especificación constante puede ser:

- *Vacío.* $\mathcal{CS} = \emptyset$, que corresponde con un agente totalmente escéptico, es decir, el agente no acepta de manera axiomática ninguna afirmación. $\mathcal{J}_\emptyset$ es la lógica de justificación que corresponde con este caso.

- *Finito.* $\mathcal{CS}$ es un conjunto finito de fórmulas, donde un número finito de axiomas y su justificación forman parte de la especificación constante. De esta manera se expresa que para una derivación en el lenguaje de la lógica de justificación se utilice un número de pasos finito, de acuerdo con las reglas de inferencia. A cada axioma que se considere válido en una lógica dada, le corresponde una constante de justificación.

- *Axiomáticamente apropiado.* Esta caracterización corresponde con que, para un axioma $A$, y una constante de justificación $e_1$, tal que $e_1 : A \in \mathcal{CS}$, existe una justificación $e_2$ adicional, tal que $e_2 : e_1 : A \in \mathcal{CS}$, y en general, si la secuencia de términos de justificación $e_n : e_{n-1} : \ldots : e_1$, es tal que

$$e_n : e_{n-1} : \ldots : e_1 : A \in \mathcal{CS},$$

entonces

$$e_{n+1} : e_n : e_{n-1} : \ldots : e_1 : A \in \mathcal{CS}$$

Es decir, para todo axioma $A$ en el conjunto de especificación constante, existe un término de justificación adicional $e_{n+1}$, para cada miembro de $\mathcal{CS}$. Esto garantiza que se cumpla el principio de internalización.

- *Total.* Un conjunto de especificación constante es total si para todo axioma $A$, debe existir un conjunto de constantes de justificación $e_1, e_2, \ldots, e_n$, tal que se cumple

$$e_n : e_{n-1} : \ldots : e_1 : A \in \mathcal{CS}$$

es decir, para todo axioma $A$, por considerarse verdadero sin verificación previa, se puede encontrar una secuencia de longitud arbitraria de constantes de justificación.

De esta manera, se puede clasificar a una lógica de justificación por su conjunto de especificación constante, es decir, los axiomas que se consideran verdaderos y los términos de justificación asociados.

Una lógica de justificación $\mathcal{J}_{\mathcal{CS}}$, estará dada por:

$$\mathcal{J}_{\mathcal{CS}} = \mathcal{J}_{\emptyset \cup \mathcal{CS}}$$

En particular, **LP** es la versión justificada de la lógica modal **S4**, caracterizada por aceptar los axiomas: $t : \varphi \rightarrow \varphi$, $t : \varphi \rightarrow !t : t : \varphi$, y $s : (\varphi \rightarrow \psi) \rightarrow t : \varphi \rightarrow (s \cdot t) : \psi$, que corresponden a las versiones explícitas de **T**, **4** y **K**, respectivamente, en **S4**, cada axioma tendrá asociada una constante de justificación.

Sea $cs\mathcal{S}4$ el conjunto de especificación constante de una versión explícita de **S4**, si consideramos la lista de axiomas dada en el apéndice § A, de A.3.1 a A.3.9, el contenido de este conjunto sería:

$$cs\mathcal{S}4 = \{e_1 : \mathit{Then}1, \ldots, e_{10} : (t : \varphi \rightarrow \varphi), e_{11} : (t : \varphi \rightarrow !t : t : \varphi), e_{12} : (s : (\varphi \rightarrow \psi) \rightarrow t : \varphi \rightarrow (s \cdot t) : \psi)\}$$

Que sería la descripción del conjunto de especificación constante (finito) de **LP**.

$$\mathbf{LP} = \mathcal{J}_{\emptyset \cup cs\mathcal{S}4}$$

**Internalización**

Los axiomas lógicos se consideran justificados sin hacer una verificación adicional, para expresar eso se utilizan los términos de justificación constantes. Esta idea está relacionada con la noción de consecuencia lógica y con el teorema de deducción, los problemas e implicaciones de esta noción de inferencia pueden ser consultados en trabajos como (Hakli & Negri, 2012; Hawthorn, 1990). La idea en este axioma es que, si se considera una afirmación verdadera, sin lugar a dudas, también se sabe que existe una justificación constante (que no requiere verificación) para ello, sea o no que la conozca el agente cognitivo. Este axioma simplemente afirma que la justificación existe, es decir:

*Para cada axioma $A$, $c{:}A$ es de nuevo un axioma, con $c$ una justificación constante, es decir*

$$\frac{A}{c{:}A}$$

Es una forma explícita de la regla de necesitación modal (**NEC**), que permite razonar sobre la longitud de la demostración. Poder establecer relaciones sobre la longitud de la demostración es una de las diferencias fundamentales y la principal ventaja con respecto a la lógica epistémica convencional, en la que hacer esto no es posible por ser justificaciones implícitas que dependen de un elemento fuera del lenguaje, como la relación de accesibilidad del agente a los mundos posibles.

Un axioma en lógica de justificación se reconoce por ser miembro del conjunto de especificación constante $\mathcal{CS}$, de la lógica que se discuta. Es decir, una fórmula $A$, con su justificación constante $c$, se considera un axioma si forma parte de $\mathcal{CS}$, es decir, $c{:}\, A \in \mathcal{CS}$.

**Conocimiento fáctico, axioma** T

El principio de conocimiento fáctico (*cf. factivity*) establece que si una proposición constituye conocimiento, esa proposición debe ser verdadera; en lógica epistémica a esto se le conoce como *axioma de verdad*, **T**:

$$KF \rightarrow F$$

En matemáticas, esto representa que si la fórmula $F$ tiene una demostración, se puede considerar verdadera. Esta propiedad del conocimiento tiene que ver con que, si un agente considera cierta una afirmación epistémica en todos los mundos accesibles, esa afirmación constituye conocimiento y debe ser verdadera.

La lógica de justificación no requiere de aceptar el axioma **T**, lo que hace posible hablar de suposiciones justificadas parcialmente, sin que ello signifique la verdad de la afirmación; o justificaciones fácticas, que cumplan con lo que estipula **T**. Si se acepta este axioma, su versión explícita quedaría como:

Para un término de justificación $t$, y una fórmula $F$ se tiene:

$$t{:}\, F \rightarrow F$$

## 2.3. Semántica

La semántica estándar para esta lógica es una adaptación de Melvin Fitting (Fitting, 2005) de los modelos de Kripke de mundos posibles, ya convencionales para definir semánticas de lógica modal e intuicionista.

2.3.1. **Definición**. Un *modelo de Kripke-Fitting* es la tupla: $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$, que consiste en un *modelo de Kripke* $\langle \mathcal{W}, \mathcal{R}, \mathcal{V} \rangle$ y una función adicional de evidencia $\mathcal{E}$, tal que $\mathcal{E}(t, F) \subseteq \mathcal{W}$, para alguna justificación $t$ de la fórmula $F$, donde:

- $\mathcal{W}$, es el conjunto de mundos / estados posibles.
- $\mathcal{R} \subseteq \mathcal{W} \times \mathcal{W}$, la relación de accesibilidad entre pares de $\mathcal{W}$.
- $\mathcal{V}: \mathcal{W} \rightarrow \wp(PROP)$, la función de etiquetación, cuyo dominio es $\mathcal{W}$ y codominio el conjunto potencia de los átomos proposicionales, tal que $PROP = \{p_0, p_1, \ldots, p_n\}$.
- $\mathcal{E}: (Just \times Forms) \rightarrow \wp(\mathcal{W})$, la función que relaciona términos de justificación con fórmulas del lenguaje y mapea este par a subconjuntos de $\mathcal{W}$, tal que:
    - *Just* es el conjunto de términos de justificación construidos de acuerdo con (2.1.1).
    - *Forms* es el conjunto de fórmulas construidas de acuerdo con (2.1.2).

Con lo anterior ya se tienen los elementos necesarios para enunciar la semántica de la lógica de justificación.

Sea $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$ un *modelo de Kripke-Fitting*, la verdad de la fórmula $\varphi$ dada por (2.1.2), en el mundo $w \in \mathcal{W}$, que se escribe como $\mathcal{M}, w \vDash \varphi$, y está dada por:

Para cada $w \in \mathcal{W}$,

1. $\mathcal{M}, w \models p$, ssi[7] $p \in \mathcal{V}(w)$, para una variable proposicional $p$;

2. $\mathcal{M}, w \models \top$ siempre es el caso;

3. $\mathcal{M}, w \models \neg\varphi$, ssi no es el caso que $\mathcal{M}, w \models \varphi$, es decir, $\mathcal{M}, w \not\models \varphi$;

4. $\mathcal{M}, w \models (\varphi \rightarrow \psi)$, ssi $\mathcal{M}, w \models \varphi$ implica $\mathcal{M}, w \models \psi$;

5. $\mathcal{M}, w \models t{:}\varphi$, ssi $w \in \mathcal{E}(t, \varphi)$ y para cada $v \in \mathcal{W}$, tal que si $wRv$, entonces $\mathcal{M}, v \models \varphi$.

De (1) a (4), la semántica es la misma que la lógica proposicional convencional, que se basa en una función de evaluación $\mathcal{V}$ para definir la verdad de una variable proposicional. El caso interesante es (5), en el que se define la relación entre una fórmula $\varphi$ y una justificación $t$. Si en el mundo $w$, en el que está posicionado el agente, se tiene una justificación $t$ para la fórmula $\varphi$, a través de la función de evidencia $\mathcal{E}$, y en todos los mundos accesibles desde $w$ esa fórmula se satisface, entonces $t{:}\varphi$ se satisface desde el mundo $w$.

[7] «ssi» se usará como abreviatura de «si y solo si».

**Propiedades de la función $\mathcal{E}$**

Para garantizar la validez de los axiomas principales de la lógica de justificación, es necesario que cumpla las siguientes propiedades:

2.3.2. *E1*. $\mathcal{E}(s, P \to Q) \cap \mathcal{E}(t, P) \subseteq \mathcal{E}(s \cdot t, Q)$

Esta propiedad representa la situación en la que la evidencia $s$ es relevante para el agente con la fórmula $P \to Q$, y la evidencia $t$ para la fórmula $P$. De esta manera, los términos de justificación $s$ y $t$, en combinación por el operador de aplicación, son evidencia suficiente para $Q$. Como es de esperarse, esta propiedad garantiza la validez del axioma de aplicación.

*2.3.3. E2*. $\mathcal{E}(s, P) \cup \mathcal{E}(t, P) \subseteq \mathcal{E}(s + t, P)$

Esta propiedad representa la situación en la que el agente acepta $s$ o $t$ (o ambos) como evidencia relevante para la fórmula $P$. El conjunto que resulta de la unión de ambos, representado por los términos combinados por el operador de suma, también es evidencia para $P$. Esta propiedad garantiza que se cumplan los dos casos del axioma de suma.

2.3.4. *E3*. $\mathcal{E}(t, P) \subseteq \mathcal{E}(!\, t, t{:}\, P)$

Esta propiedad debe cumplirse para $\mathcal{E}$, si se acepta el axioma de verificación. Representa la situación en la que, teniendo la pieza evidencia $t$ para una fórmula $P$, existe un término de justificación adicional $!\, t$, que justifica a la fórmula $t{:}\, P$.

### LP **no es una lógica modal normal**

Una lógica modal *normal* es aquella que acepta el axioma **K**. Hasta este punto, es posible ver a la lógica de justificación como una lógica modal y los términos de justificación como modalidades. Sin embargo, dentro de los axiomas que acepta **LP**, no está el que correspondería con

$$\mathbf{K}\colon s\colon (P \to Q) \to (s\colon P \to s\colon Q)$$

para un polinomio $s$, y para las fórmulas $P$ y $Q$.

Un modelo $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{V}, \mathcal{E} \rangle$, que sirve de contraejemplo para esto es el siguiente:

$\mathcal{W} = \{1\}$; $\mathcal{R} = \{\}$; $\mathcal{V}(1) = \{P, Q\}$; $\mathcal{E}(s, P \to Q) = \{1\}$, $\mathcal{E}(s, P) = \{1\}$,

y por la propiedad *E1* (2.3.2) de $\mathcal{E}$, también se tiene que $\mathcal{E}(s \cdot s, Q) = \{1\}$. Por lo tanto, en $\mathcal{M}$ se cumple:

$\mathcal{M}, 1 \vDash s\colon (P \to Q)$

$\mathcal{M}, 1 \vDash s\colon P$

Sin embargo, esto no es suficiente para concluir $\mathcal{M}, 1 \vDash s\colon Q$, porque de acuerdo con la semántica de mundos posibles justificados, es necesario que se cumpla $\mathcal{E}(s, Q) = \{1\}$. Por lo tanto, $\mathcal{M}, 1 \nvDash s\colon Q$, esto a pesar de que tengamos evidencia para $\mathcal{M}, 1 \vDash (s \cdot s)\colon Q$, por el axioma de aplicación. Las reglas de la lógica de justificación permiten distinguir entre $s\colon Q$ y $(s \cdot s)\colon Q$, es decir, distintas justificaciones para la misma fórmula.

**Percepción lógica (*cf. Logical Awareness*)**

La noción de percepción lógica se puede encontrar desde el trabajo seminal sobre la formalización del conocimiento (Hintikka, 1962), donde Hintikka describe dos maneras en que un agente cognitivo puede alcanzar conocimiento, y lo hace con el propósito de señalar lo que se conoce hoy en día como «el problema de la omnisciencia lógica», que se tratará a detalle más adelante. En este apartado, sobre el conjunto de especificación constante, es pertinente señalar que el lenguaje de la lógica de justificación puede distinguir entre suposiciones y suposiciones justificadas, una característica muy importante cuando se habla de conocimiento en el contexto de un aparto formal. El tratamiento que da Hintikka a la percepción lógica ha inspirado desarrollos que tratan dos tipos de conocimiento, el *implícito* y el *explícito*, donde el conocimiento explícito es el que el agente cognitivo está al tanto, se ha percatado (para señalar la idea de *awareness*); y el conocimiento implícito es consecuencia lógica del explícito, es decir, es una inferencia que hace el agente con el conocimiento que posee.

En lógica de justificación, el conjunto de especificación constante permite hablar de afirmaciones que no requieren más trabajo de verificación del agente, están justificadas *ex officio*. Por ejemplo, se postula un axioma $A$ para un agente, lo que en lógica de justificación resultaría en que, para un término de justificación constante $e_1$, la fórmula $e_1 : A$ no requiere verificación. Si, por otro lado, se postula una afirmación $P$ (una fórmula del lenguaje) que el agente no considera como un axioma,

se puede enunciar a través de variables de justificación, $x : P$[8], de modo que también representa una suposición justificada, pero no se establece de antemano la relación entre el término $x$ y la fórmula $P$ (en la función de evidencia $\mathcal{E}$), es decir, el lenguaje solo expresa la idea de una justificación parcial o indeterminada, lo que no significa que la fórmula sea verdadera, lo que se decidiría analizando la fórmula semánticamente, a través de un modelo para la fórmula.

## 2.4. Proyección de olvido

Una conexión intuitiva entre las afirmaciones justificadas del lenguaje de la lógica de justificación y las afirmaciones modales usando el operador $\Box$ se puede pensar como una afirmación existencial, donde $\Box F$ se convierte en:

$$\textit{Para una } x, x : F$$

El lenguaje de la lógica de justificación no cuantifica directamente sobre términos de justificación, utiliza las operaciones definidas entre variables y constantes de justificación para formar polinomios de justificación. Los polinomios de justificación se pueden entender como operadores a la Skolem, reemplazando los cuantificadores existenciales con funciones que involucran términos de justificación.

[8] Si solo se postula la fórmula $P$ (sin justificación), la suposición es sobre la verdad de la fórmula, lo que limitaría el análisis epistémico de la situación (porque suponer $P$ verdadera nos obligaría a asignarle un término de justificación por internalización), al agregar el término de justificación indeterminado, es posible razonar sobre la afirmación $x : P$ sin suponer la verdad de la fórmula, algo problemático en lógica epistémica, relacionado con el conocimiento fáctico, *i.e.* el axioma **T**.

Por ejemplo, la fórmula

$$x{:}F \to f(x){:}Q$$

Es una skolemización de

$$\exists x(x{:}F) \to \exists y(y{:}Q)$$

Vistos de este modo, los sistemas lógicos de justificación son lógicas de creencia y conocimiento en el mismo aparato, convirtiendo los cuantificadores existenciales en funciones de justificación. En el otro sentido, para convertir fórmulas justificadas en fórmulas de lógica modal se utiliza la proyección de olvido $^{\circ}$, que reemplaza cada ocurrencia de un término de justificación $t$ en $t{:}F$ por $\Box$, es decir, $t{:}F^{\circ} \Box F$.

La fórmula

$$x{:}F \to f(x){:}Q$$

Aplicando la proyección de olvido, es una versión explícita de

$$\Box F \to \Box Q$$

Algunos ejemplos, con $p$ y $q$ como átomos proposicionales son:

$$t{:}p \quad {}^{\circ} \quad \Box p$$

$$t{:}\,p \to !\,t{:}\,t{:}\,p \quad \rightsquigarrow \quad \Box p \to \Box\Box p$$

$$s{:}\,(p \to q) \to t{:}\,p \to (s \cdot t){:}\,q \quad \rightsquigarrow \quad \Box(p \to q) \to \Box p \to \Box q$$

La proyección de olvido conserva la validez de una fórmula en lógica de justificación al mapearla a una fórmula válida de lógica modal, es decir, si una fórmula $x{:}\,P$ es válida, su proyección de olvido $\Box P$ también será válida.

Sin embargo, las versiones implícitas pueden perder información relevante en su contraparte explícita. Por ejemplo, una fórmula trivialmente verdadera $x{:}\,P \to x{:}\,P$, el axioma suma, que refleja la importante propiedad de monotonía, $x{:}\,P \to (x + y){:}\,P$; y una fórmula no válida (por no ser verdadera si $x \neq y$) $x{:}\,P \to y{:}\,P$, todas tienen la misma proyección de olvido, $\Box P \to \Box P$.

En el otro sentido también se cumple, y una fórmula válida en lógica modal es una proyección de olvido de una fórmula válida en lógica de justificación. Para los ejemplos anteriores, la fórmula modal válida $\Box P \to \Box P$, pudo venir de las fórmulas $x{:}\,P \to x{:}\,P$, o $x{:}\,P \to (x + y){:}\,P$ que son válidas y se conserva la validez de las fórmulas. En el caso de $x{:}\,P \to y{:}\,P$, por no ser verdadera si $x \neq y$, aunque se puede obtener su proyección de olvido (que sería $\Box P \to \Box P$), no hay validez que preservar. Lo que no puede ocurrir es que a partir de una fórmula válida $x{:}\,P$ se obtenga una proyección de olvido $\Box P$ que no lo sea.

Extendiendo la idea de la proyección de olvido a lógicas completas, Artemov en (S. Artemov, 2008) describe el proceso para obtener versiones explícitas de sistemas epistémicos comunes.

## 2.5. Ejemplos de derivaciones en LP

El capítulo «Conocimiento y justificación» se dedicará al análisis y formalización de casos problemáticos de conocimiento como los propuestos por Gettier. En este apartado se dan dos ejemplos de derivación para ilustrar el poder de la lógica de justificación.

### Granero rojo en LP

La siguiente derivación se presenta originalmente en (S. Artemov & Fitting, 2012), que utiliza la versión simplificada (Dretske, 2005) del ejemplo del *Granero Rojo* de Goldman y Kripke, donde se discuten las implicaciones de que el conocimiento sea cerrado bajo consecuencia epistémica:

> *"Suppose I am driving through a neighborhood in which, unbeknownst to me, papier-mâché barns are scattered, and I see that the object in front of me is a barn. Because I have barn-before-me percepts, I believe that the object in front of me is a barn. Our intuitions suggest that I fail to know barn. But now suppose that the neighborhood has no fake red barns, and I also notice that the object in front of me is red, so I know a red barn is*

> *there. This juxtaposition, being a red barn, which I know, entails there being a barn, which I do not, 'is an embarrassment'."*[9] *(Dretske, 2005)*

La primera formalización considera una modalidad que se interpreta como «creencia», es decir, el enunciado $\Box P$, se interpreta como «el agente cree en la afirmación $P$», de la siguiente manera:

1. $\Box B$, «creo que el objeto frente a mí es un granero»;

2. $\Box(B \wedge R)$, «creo que el objeto frente a mí es un granero rojo».

Desde un punto de vista externo[10], 2. $\Box(B \wedge R)$ es de hecho conocimiento y no mera creencia, y 1. $\Box B$, es solo una creencia para el agente porque hay graneros potencialmente falsos.

3. $\Box(B \wedge R \rightarrow B)$ , es una afirmación modal de conocimiento por ser $(B \wedge R \rightarrow B)$ una tautología, es decir, se cumple para todo mundo accesible.

[9] Imagine que estoy conduciendo en un vecindario en el cual, sin que yo lo sepa, hay graneros falsos a lo largo del camino, y veo que el objeto frente a mí es un granero. Debido a que cuento con la capacidad de percepción para identificar graneros, formo la creencia de que el objeto frente a mí es un granero. Nuestras intuiciones sugieren que fallo en afirmar que sé que es un granero. Pero ahora suponga que en el mismo vecindario no hay graneros falsos rojos, y además noto que el objeto frente a mí es rojo, por lo que sé que un granero rojo está ahí. Esta yuxtaposición, de ser un granero rojo que conozco, que implica que es un granero que no conozco, 'es vergonzoso'.

[10] Es decir, es posible juzgar la situación desde una posición privilegiada, externa a los agentes cognitivos, que permite decidir si se trata de mera creencia o de conocimiento.

Sin embargo, en (3), si se analiza desde un punto de vista externo, no parece posible una derivación válida de conocimiento, debido a que (1) es solo una creencia y la conclusión se considera conocimiento.

Ahora consideremos el mismo ejemplo con lógica de justificación y la interpretación de «$t{:}\,P$» como «$t$ es una razón para creer $P$».

1. $u{:}\,B$, «$u$ es una razón para creer que el objeto frente a mí es un granero»;

2. $v{:}\,(B \wedge R)$, «$v$ es una razón para creer que el objeto frente a mí es un granero rojo»;

3. $a{:}\,(B \wedge R \rightarrow B)$, como en el caso anterior, «$B \wedge R \rightarrow B$» es una tautología, por lo que $a$ es un justificación constante para la fórmula.

Similar al caso anterior, (2) y (3) constituyen afirmaciones de conocimiento, pero (1) no, por existir posibles graneros falsos. Desde un punto de vista externo es posible derivar solo desde (2) y (3) lo siguiente:

4. $a{:}\,(B \wedge R \rightarrow B) \rightarrow (v{:}\,(B \wedge R) \rightarrow (a \cdot v){:}\,B)$, se satisface por el axioma de Aplicación en conjunto con (2) y (3);

5. $v{:}\,(B \wedge R) \rightarrow (a \cdot v){:}\,B$, por el paso (3) y (4) y *Modus Ponens* de lógica proposicional;

6. $(a \cdot v){:}\,B$, por (2) y (5) y otra aplicación de *Modus Ponens* de lógica proposicional.

La conclusión (6) es distinta a (1) en el lenguaje de la lógica de justificación, pues aunque es la misma afirmación $B$ (el objeto frente a mí es un granero), la justificación es distinta, es decir, $(a \cdot v) : B$ es una afirmación distinta a $u : B$. La capacidad de hacer esta distinción hace posible validar afirmaciones que constituyen creencias y diferenciarlas del conocimiento, algo que no es posible en una lógica modal epistémica convencional.

Aquí hay que notar que esta distinción no tiene que ver con la validez de la justificación, únicamente que es posible expresar formalmente estos casos, en donde es problemático diferenciar creencia y conocimiento. En este caso, no sabemos (ni es el objeto de la formalización) qué hace diferentes a las justificaciones $(a \cdot v)$ y $u$; en el lenguaje de lógica de justificación son términos de justificación distintos que vienen de pasos válidos en la inferencia.

Es aquí donde se revela el poder de esta lógica, que permite razonar sobre la justificación de afirmaciones de conocimiento y creencia en un mismo aparato formal. Poder derivar casos como estos no resuelve dilemas filosóficos o epistemológicos, pero proporciona al investigador una herramienta que puede utilizar para analizar sus afirmaciones, de modo similar a como se utiliza la lógica proposicional o la lógica modal. La propiedad de cerradura de afirmaciones de conocimiento es importante también porque evita que se concluyan afirmaciones de conocimiento de forma incorrecta por venir de premisas que constituyan creencia sin ser conocimiento.

**Ejemplos de Gettier: Caso I**

La siguiente formalización es el Caso I (Gettier, 1963), que presenta Gettier para señalar lo que considera un problema en la definición de conocimiento como creencia verdadera justificada.

> *Smith se postula para un trabajo, pero tiene la creencia justificada de que «Jones será elegido para el puesto». También tiene la creencia justificada de que «Jones tiene 10 monedas en su bolsa». Smith, por lo tanto, puede concluir de manera justificada que… «el hombre que consiga el trabajo tiene 10 monedas en la bolsa».*
>
> *Pero sucede que Jones no consigue el trabajo, sino Smith. Sin embargo, resulta que Smith tiene 10 monedas en la bolsa. Por lo que su creencia de que «el hombre que consiga el trabajo tiene 10 monedas en la bolsa» estaba justificada y resulta cierta. Pero esto no parece capturar la noción de conocimiento.*

Los ejemplos que plantea Gettier presentan casos de creencias verdaderas justificadas, pero no parecen ser conocimiento.

Usando lógica de justificación, Artemov en (S. Artemov, 2008) formaliza este caso de Gettier para argumentar que no se trata de conocimiento, es decir, Smith no tiene evidencia adecuada para que su afirmación constituya conocimiento.

Esta formalización funciona como una defensa a la definición de creencia verdadera justificada para el conocimiento. Primero se hará un análisis semántico con modelos de Kripke-Fitting para ofrecer un contraejemplo donde Smith no tiene

evidencia suficiente para concluir que su afirmación constituye conocimiento. Posteriormente, se hará un análisis sintáctico utilizando las mismas premisas de esta formalización, pero como se verá, para hacer esta derivación será necesario agregar una premisa adicional.

Se utilizarán las siguientes variables proposicionales:

- *JJ*: Jones consigue el trabajo;
- *SJ*: Smith consigue el trabajo;
- *JC*: Jones tiene 10 monedas en el bolsillo;
- *SC*: Smith tiene 10 monedas en el bolsillo;
- *x*: evidencia que Smith tiene sobre *JJ*;
- *y*: evidencia que Smith tiene sobre *JC*.

El caso en concreto plantea las siguientes premisas:

1. $x : JJ$ (*x* es una justificación para «Jones consigue el trabajo»);
2. $y : JC$ (*y* es una justificación para «Jones tiene 10 monedas en el bolsillo»);
3. $\neg JJ$ (Jones no consigue el trabajo);
4. *SJ* (Smith consigue el trabajo);
5. *SC* (Smith tiene 10 monedas en el bolsillo).

El argumento completo:

$$x{:}\,JJ, y{:}\,JC, \neg JJ, SJ, SC \vdash t{:}\,[(JJ \rightarrow JC) \wedge (SJ \rightarrow SC)]$$

Lo que hay que demostrar es que con esas suposiciones no se puede garantizar la verdad de

$$t{:}\,[(JJ \rightarrow JC) \wedge (SJ \rightarrow SC)]$$

es decir, que Smith tiene una justificación para en creer que quien tenga 10 monedas en el bolsillo, obtiene el trabajo.

***Análisis semántico***

Basta dar un modelo en el que se cumplan las premisas pero no la conclusión:

Sea $\mathcal{W} = \{1,2\}$, $\mathcal{R} = \{(1,2)\}$, y para cada mundo:

$$1 \models SJ, SC, JC, \neg JJ$$

$$2 \models JJ, JC, SJ, \neg SC$$

(1) y (2) representan dos posibles situaciones que se pueden presentar. (1) es la situación «real» en la que Smith obtiene el trabajo (*SJ*), tiene 10 monedas (*SC*), Jones también tiene 10 monedas (*JC*) y no consigue el trabajo ($\neg JJ$); y (2) es una situación en la que Jones consiguió el trabajo (*JJ*), Jones tienen 10 monedas (*JC*), y Smith no tiene las 10 monedas ($\neg SC$).

La situación (2) está así planteada para exponer un ejemplo en el que se satisfacen las premisas del argumento pero no la conclusión, por lo que el agente (Smith),

con las premisas de este caso y razonando con justificaciones, no tiene suficiente información para concluir que su afirmación «quien tenga diez monedas en la bolsa conseguirá el trabajo».

$\mathcal{E}$ está dada por:

$$\mathcal{E}(x, JJ) = \{1, 2\}$$

$$\mathcal{E}(y, JC) = \{1, 2\}$$

Ahora procedemos a evaluar en el mundo 1 con las especificaciones del modelo. Para el mundo 1:

1. $x{:}\,JJ$, se cumple porque $1 \in \mathcal{E}(x, JJ)$, y por $(1,2) \in \mathcal{R}$, también se cumple $2 \vDash JJ$

2. $y{:}\,JC$, se cumple porque $1 \in \mathcal{E}(y, JC)$, y por $(1,2) \in \mathcal{R}$, también se cumple $2 \vDash JC$

3. $\neg JJ$, se cumple trivialmente por $1 \vDash \neg JJ$

4. $SJ$, se cumple trivialmente por $1 \vDash SJ$

5. $SC$, se cumple trivialmente por $1 \vDash SC$

Pero, $t{:}\,[(JJ \rightarrow JC) \wedge (SJ \rightarrow SC)]$ no se cumple para algún término de justificación $t_1$ porque $[(JJ \rightarrow JC) \wedge (SJ \rightarrow SC)]$ es falsa en (2), por $2 \vDash SJ$, pero $2 \vDash \neg SC$, y $(1,2) \in \mathcal{R}$.

■

***Análisis sintáctico***

Adicional a las premisas (1) a (5), pareciera que Gettier pretendía en el Caso 1 dar la idea de que solo uno de los dos postulantes conseguiría el trabajo, es decir, se puede integrar la siguiente premisa al argumento:

6. $z$: $(JJ \rightarrow \neg SJ)$, $z$ es una justificación para «si Jones consigue el trabajo, entonces Smith no»

7. $w$: $(SJ \rightarrow \neg JJ)$, $w$ es una justificación para «si Smith consigue el trabajo, entonces Jones no»

hay que notar que la afirmación tiene la justificación variable $z$, que representa la razón del agente (Smith) para creer esta afirmación.

La derivación es como sigue:

8. $(z \cdot x)$: $\neg SJ$, de (1) y (6), usando el axioma de Aplicación;

9. $p$: $(\neg SJ \rightarrow (SJ \rightarrow SC))$, Internalización de una tautología, para una constante de justificación $p$;

10. $(p \cdot (z \cdot x))$: $(SJ \rightarrow SC)$, Aplicación (8, 9);

11. $c$: $(JC \rightarrow (JJ \rightarrow JC))$, Internalización de una tautología, para la constante $c$;

12. $(c \cdot y)$: $(JJ \rightarrow JC)$, Aplicación (2, 11);

13. $t$: $[(JJ \rightarrow JC) \wedge (SJ \rightarrow SC)]$, para un polinomio de justificación $t$, de (10) y (12), con $t = (d \cdot (p \cdot (z \cdot x))) \cdot (c \cdot y)$, a partir de las siguientes fórmulas justificadas por constantes:

- $c$: $(JC \rightarrow (JJ \rightarrow JC))$,
- $p$: $(\neg SJ \rightarrow (SJ \rightarrow SC))$,
- $d$: $(SJ \rightarrow SC) \rightarrow ((JJ \rightarrow JC) \rightarrow ((SJ \rightarrow SC) \wedge (JJ \rightarrow JC)))$.

■

En el paso (13), el cálculo del polinomio $t$ parece complicado, pero se puede ver como el resultado obtener la elucidación de la fórmula modal:

$$\Box\varphi \wedge \Box\psi \rightarrow \Box(\varphi \wedge \psi)$$

que resulta en una fórmula justificada con la forma, $x{:}\,\varphi \wedge y{:}\,\psi \rightarrow t{:}\,(\varphi \wedge \psi)$. Para aclarar estas operaciones, se calculará la forma general de un polinomio $t$ de justificación partiendo de la conjunción de dos fórmulas justificadas con las variables de justificación $x$ y $y$.

13.1. $\varphi \rightarrow (\psi \rightarrow (\varphi \wedge \psi))$, una tautología

13.2. $d{:}\,(\varphi \rightarrow (\psi \rightarrow (\varphi \wedge \psi)))$, Internalización de (13.1) con la constante $d$

13.3. $d{:}\,(\varphi \rightarrow (\psi \rightarrow (\varphi \wedge \psi))) \rightarrow x{:}\,\varphi \rightarrow (d \cdot x){:}\,(\psi \rightarrow (\varphi \wedge \psi))$, instancia de axioma de Aplicación

13.4. $x{:}\,\varphi \rightarrow (d \cdot x){:}\,(\psi \rightarrow (\varphi \wedge \psi))$, *Modus Ponens* (13.2, 13.3)

13.5. $(d \cdot x){:}\,(\psi \to (\varphi \wedge \psi)) \to (y{:}\,\psi \to ((d \cdot x) \cdot y){:}\,(\varphi \wedge \psi))$, instancia de Aplicación

13.6. $x{:}\,\varphi \to (y{:}\,\psi \to ((d \cdot x) \cdot y){:}\,(\varphi \wedge \psi))$, silogismo hipotético (13.4, 13.5)

13.7. $x{:}\,\varphi \wedge y{:}\,\psi \to ((d \cdot x) \cdot y){:}\,(\varphi \wedge \psi)$, tautología proposicional equivalente a (13.6)

Con la forma general de este polinomio, sustituyendo el valor de $\varphi$ y $\psi$ en el esquema y los valores de las variables de justificación, $x = (p \cdot (z \cdot x))$, y $y = (c \cdot y)$, se obtiene el resultado de la derivación anterior.

Hasta aquí parece todo correcto, y se puede tomar como un ejemplo para entender las derivaciones con una lógica de justificación. Sin embargo, tener una fórmula con términos de justificación, no garantiza el conocimiento, que es precisamente lo que reflejan los casos Gettier o el ejemplo del granero rojo. Si aceptamos el principio de conocimiento fáctico (axioma **T**) en la lógica de justificación, como suele hacerse en lógica epistémica, podríamos agregar la siguiente premisa al argumento original.

6'. $x{:}\,JJ \to JJ$

donde solo sustituimos a $x$ por el término de justificación arbitrario y $JJ$ en el axioma **T** explícito. Agregar esta premisa resulta en una contradicción aplicando Modus Ponens con la premisa (1), lo que resulta en $JJ$, que se contradice con (3),

$\neg JJ$. El significado de esta contradicción es que las condiciones del ejemplo propuesto por Gettier no cumplen con los requisitos para establecer que la creencia de Smith es conocimiento, a pesar de contar con una justificación.

Cuando se formalizan los casos Gettier y otros ejemplos similares, necesariamente se extrae información particular que no se incluye en el lenguaje formal deductivo. Una formalización no podría responder a los intereses o deseos de los agentes (o de cualquier elemento que no esté expresado en el lenguaje). El éxito de una formalización está en elucidar información relevante para quien la utiliza. El principal problema de estos ejemplos, no es que las justificaciones fallen, sino que el razonamiento modal solo permite trabajar con afirmaciones implícitas. Es decir, un razonamiento modal solo puede tratar con afirmaciones que tienen o no justificación, pero no podemos desenvolver el carácter de esa justificación. Una formalización como la lógica de justificación puede contribuir a mejorar el análisis que se hace de estos casos, permitiendo hacer un diagnóstico más granular de las justificaciones.

# 3. Conocimiento y justificación

## 3.1. Introducción

La utilidad de tener mecanismos automáticos para resolver problemas la vemos en el desarrollo tecnológico de la actualidad. Aplicar estos mecanismos a situaciones epistémicas ha sido el objetivo de la inteligencia artificial desde sus inicios. Turing imaginó máquinas que pudieran tener conversaciones con seres humanos sin que éstos notaran que se trataba de una máquina (Turing, 1950). El problema de hablar de actitudes epistémicas en cualquier contexto es que no podemos ignorar las discusiones sobre lo que es y no es la mente, el pensamiento o el conocimiento. La relevancia de los aparatos formales en estas discusiones es que, al poder describir las operaciones para realizar una tarea, sería posible, al menos en principio, codificar información e introducirla a un dispositivo mecánico (una computadora) para que un realice esa actividad.

En este capítulo se revisarán alternativas para representar el conocimiento en diferentes ámbitos. En particular, profundizaremos en el enfoque de la lógica epistémica tradicional, por ser el marco más utilizado en lógica y en computación. La idea es revisar la noción de justificación en lógica modal epistémica y compararla con la lógica de justificación, que permite expresar razones para nociones epistémicas.

Como el análisis debe ser lo más general posible, el concepto de *noción epistémica* debe abarcar tanto a la *creencia* como al *conocimiento*. La lógica de justificación tam-

bién debe tomarse como un aparato formal para operar sobre estas nociones epistémicas generales. La *justificación*, entendida de manera general, representa una idea de evidencia, razón, relevancia, explicación, etc., aplicada a fórmulas del lenguaje. La interpretación de los términos de justificación sería entonces análoga a las modalidades en lógica modal, así como tenemos lógicas: deónticas, doxásticas, temporales, dinámicas, de demostración, etc.; en lógica de justificación tendríamos términos de justificación entendidos como: evidencia, relevancia, no contradicción, verificación, demostración, etc., de acuerdo al contexto donde se utilicen.

Al final de este capítulo se presentarán ejemplos de situaciones en las que la lógica de justificación puede ser útil para mejorar el análisis de situaciones epistémicas.

## 3.2. Definición de conocimiento

La definición tradicional de conocimiento en epistemología (Dancy & Celma, 1993, p. 39) para un agente *S*, y una afirmación *P*, es la siguiente:

Un agente *S* sabe / conoce *P* si:

1. *S* cree en *P*;
2. *S* está justificado en creer en *P*;
3. *P* es verdad.

Donde (1) establece que el agente pueda formar una *creencia* de la afirmación, (2) que el agente debe *tener una razón para creer* en la afirmación, y (3) que esa afirmación debe ser *verdadera*.

Esta noción de conocimiento es útil para analizar casos que involucren afirmaciones epistémicas. El conocimiento como «creencia verdadera justificada» captura la generación de una afirmación acerca de algo (la creencia), un esquema de validación de esa afirmación (la verdad) y una razón para esa afirmación (la justificación). Esta definición puede no ser la más popular[11], pero es suficientemente general y permite entender muchas cosas por creencia o justificación, y pensar al conocimiento (en términos generales) como esa noción tripartita. Los casos Gettier (Gettier, 1963) señalan problemas con esta definición en cuanto a que no son las condiciones suficientes, aunque sí son necesarias para el conocimiento. Es decir, incluso para Gettier algo que es conocimiento puede ser expresado como una creencia verdadera justificada, y el problema que señala es que esta definición puede incluir cosas que no son conocimiento.

### 3.3. Representación del conocimiento

El deseo de entender y reproducir comportamientos racionales llevó a desarrollar estrategias para hablar de conocimiento. Además de la lógica epistémica que se usa en computación y en filosofía, la inteligencia artificial se ha encargado de sofisticar estas estrategias añadiendo elementos distintos al procesamiento simbólico de proposiciones.

[11] Esta definición no captura, por ejemplo, el uso de la palabra conocimiento del tipo «José conoce a Pedro» o «saber andar en bicicleta». Esta idea tripartita de conocimiento tiene que ver más con el análisis proposicional del conocimiento.

La inteligencia artificial nació con el propósito de emular el pensamiento humano en un mecanismo que permitiera decidir de forma automática problemas que requieren de participación humana. La idea del pensamiento automatizado condujo a desarrollar tecnología que capturara la manera de pensar de lo que se considera comportamiento racional. El ser humano es un ejemplo de comportamiento racional, y la idea principal era hacer eso que hacen los humanos pero más rápido y mejor. Para poder emular el pensamiento humano, la lógica es nuestra herramienta más poderosa, porque permite formular reglas racionales a través de símbolos y operaciones bien definidas.

El enfoque de utilizar lógica para representar conocimiento, en principio sonaba alentador en el proyecto de representar comportamiento racional. La lógica en inteligencia artificial, a través de un lenguaje formal y la aplicación de reglas de inferencia, buscaba dar una notación precisa para analizar afirmaciones de un agente racional, pero esto es solo una parte del comportamiento racional. No siempre es necesario hacer una inferencia[12] debido a que puede no haber una razón suficiente para decidir, o tiempo suficiente, o simplemente no sea tratable el problema. En estos casos, el agente racional (emulando al ser humano) reacciona de acuerdo a otros criterios (p. ej. optimización). Por ejemplo, cuando de manera súbita nuestras

[12] En (Russell & Norvig, 2003), Russell y Norving distinguen entre razonamiento y comportamiento al reproducir pensamiento. Por mucho tiempo se concentraron los esfuerzos en el razonamiento, que llevó a desarrollar habilidades de procesamiento de lenguaje, representación de conocimiento o razonamiento automatizado. La diferencia con el enfoque en el comportamiento, es que se trata de emular la naturaleza imperfecta de los rasgos humanos al actuar, lo que (a veces) conduce a comportamiento «inteligente».

manos se encuentran con una fuente intensa de calor, normalmente es más provechoso retirarlas para evitar quemaduras, sin tener que hacer una inferencia sobre lo que hay que hacer. Un agente racional debe incluir ambas maneras de reaccionar, tanto a través de reglas de inferencia como a modo de reflejo. Un agente racional debe ser capaz de actuar de modo «inteligente» ante estas situaciones, o al menos ese era el objetivo inicial de la inteligencia artificial.

Este modo de ver el comportamiento racional finalmente hizo que la práctica de la inteligencia artificial se enfocara en acciones del agente, más que en inferencias y aparatos lógicos internos. Aquí podemos ubicar la principal diferencia entre lo que estudia la inteligencia artificial y la lógica epistémica; en la primera es importante tomar en cuenta todos los factores que intervienen en el comportamiento racional, mientras que en la segunda, se analiza un agente razonador ideal para operar con las nociones de conocimiento.

Se considera que un agente es un razonador ideal si suponemos que cuenta con los recursos (p. ej. tiempo, espacio) necesarios para realizar una inferencia o computación. Sin embargo, es útil tomar como punto de partida a un razonador de este tipo como un recurso metodológico para hacer el análisis de situaciones que involucren conocimiento. Considerar un agente de este tipo deriva en aceptar características no deseables como la omnisciencia lógica, esto se verá con más detalle en el capítulo § 4.

### 3.4. Lógica epistémica como modelo del conocimiento

En *Knowledge and Believe* (Hintikka, 1962), Hintikka explícitamente se deslinda de una definición particular de conocimiento, al argumentar que su formalización se

basa en una noción de consistencia de un conjunto de oraciones (*cf. sentences*) y afirmaciones (*cf. statements*) para expresar conocimiento o creencia de manera indistinta. Esto lo hace con el propósito de presentar un aparato formal (simbólico) para trabajar con este tipo de nociones sin lidiar con la carga filosófica que implica definir el conocimiento. No pretende dar una teoría del conocimiento, sino un aparato lógico para trabajar con afirmaciones que se pueden interpretar como conocimiento. Pero incluso en este lenguaje se pueden encontrar elementos que pueden empatar con una definición tripartita de conocimiento.

Hintikka ofrece un aparato formal que sentó las bases de lo que hoy se conoce como lógica epistémica. En esta lógica se pueden expresar afirmaciones de conocimiento y creencia, formalizando expresiones como «el agente sabe $\varphi$», «el agente cree $\varphi$», o $K\varphi$ y $B\varphi$, respectivamente. Por ejemplo,

$K_a\varphi$ se lee «el agente *a* sabe $\varphi$»,

$B_c\varphi$ se lee «el agente *c* cree $\varphi$».

Desarrollos más recientes de estas formalizaciones permiten razonar sobre más de un agente en entornos multimodales, con una modalidad para cada agente, para modelar situaciones de transferencia de conocimiento, comunicación, etc.

Normalmente se presenta con una semántica de mundos posibles y estructuras comunes en lógica modal como los *modelos de Kripke*. Dando esta interpretación de mundos posibles, tenemos:

$K_a\varphi$, significa que para todos los mundos posibles compatibles con *lo que sabe* el agente *a*, $\varphi$ es el caso;

$B_c\varphi$, significa que para todos los mundos posibles compatibles con *lo que cree* el agente *c*, $\varphi$ es el caso.

Una semántica de mundos posibles para una lógica epistémica proposicional con un solo agente *a*, consiste en un marco $\mathcal{F}$, que es una tupla $\langle \mathcal{W}, \mathcal{R}_a \rangle$, con $a \in \mathcal{A}$, donde *a* es un agente que pertenece a un conjunto de agentes $\mathcal{A}$; $\mathcal{W}$ es un conjunto no vacío de mundos posibles; $\mathcal{R}_a \subseteq \mathcal{W} \times \mathcal{W}$, es una relación binaria entre mundos desde el punto de vista de *a*. Un modelo $\mathcal{M}$, es una tupla $\langle \mathcal{W}, \mathcal{R}_a, \mathcal{L} \rangle$, formada por un marco y una función de etiquetación $\mathcal{L}: \mathcal{W} \rightarrow \wp(Atoms)$ ,que *etiqueta* cada mundo con un subconjunto de átomos proposicionales. Si un átomo proposicional *p* está etiquetado por $\mathcal{L}$, se puede leer como una asignación de verdad para *p* en el mundo correspondiente. Las fórmulas se evalúan respecto a un mundo posible, es decir, son verdaderas respecto al conjunto de mundos posibles accesibles por el agente desde un mundo / situación / estado actual. Las fórmulas están dadas por la gramática:

$$\varphi ::= \quad p \mid \top \mid \neg\varphi \mid \varphi \rightarrow \varphi \mid K_a\varphi$$

y de manera análoga para $B_a\varphi$.

El modelo $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}_a, \mathcal{L} \rangle$ se conoce como *modelo de Kripke*, que junto con la *semántica de Kripke* de mundos posibles establece la verdad de una fórmula $\varphi$ en un mundo $w \in \mathcal{W}$, para el modelo $\mathcal{M}$.

1. $\mathcal{M}, w \models p$, ssi $p \in \mathcal{L}(w)$, para una variable proposicional $p$;

2. $\mathcal{M}, w \models \top$ siempre es el caso;

3. $\mathcal{M}, w \models \neg\varphi$, ssi no es el caso que $\mathcal{M}, w \models \varphi$, es decir, $\mathcal{M}, w \not\models \varphi$;

4. $\mathcal{M}, w \models \varphi \rightarrow \psi$, ssi $\mathcal{M}, w \not\models \varphi$ o bien $\mathcal{M}, w \models \psi$;

5. $\mathcal{M}, w \models K_a\varphi$, ssi para todo $u \in \mathcal{W}$, con $wR_au$, $\mathcal{M}, u \models \varphi$.

Al ser $K$ una modalidad análoga a la necesidad, esta manera de definir la consecuencia lógica permite definir también una modalidad dual de posibilidad, tal que

6. $\mathcal{M}, w \models P_a\varphi$, ssi existe un $u \in \mathcal{W}$, con $wR_au$, $\mathcal{M}, u \models \varphi$

Esta modalidad dual representa una noción de posibilidad de que $\varphi$ sea verdadera desde el punto de vista de $a$. Hintikka representa esta modalidad como $P$, es decir, $P_a\varphi$, significa «el agente $a$ considera posible $\varphi$».

Para una lógica de creencia, la semántica es la misma de (1) a (4), la modalidad usada se representa como $B$ (*cf. Belief*), y su dual $C$, donde $B_a\varphi$ significa «el agente $a$ cree $\varphi$», y $C_a\varphi$ significa «para el agente $a$, $\varphi$ es compatible con sus creencias» y su semántica formal está dada por:

5'. $\mathcal{M}, w \models B_a\varphi$, ssi para todo $u \in \mathcal{W}$, con $wR_au$, $\mathcal{M}, u \models \varphi$.

6'. $\mathcal{M}, w \models C_a\varphi$, ssi existe un $u \in \mathcal{W}$, con $wR_au$, $\mathcal{M}, u \models \varphi$

Las modalidades en la formalización de Hintikka sugieren que se debe distinguir entre $P$ y $B$ en el análisis epistémico. La modalidad $P$ puede verse como una noción débil de creencia. Débil en el sentido de que no se considera la disposición o la intención del agente a creer en la afirmación $\varphi$ para establecer que se trata de creencia, por no corresponder con la idea habitual de creencia. Es decir, $B_a\varphi \not\equiv \neg K_a \neg \varphi$. La idea habitual de creencia requiere una especie de motivación por parte del agente para generarla, algo distinto a solo considerar posible el hecho. De acuerdo con Hintikka (Hintikka, 1962, pp. 16–19), este elemento adicional es extralógico, y se refiere a una noción de consistencia entre las afirmaciones que considera posibles el agente cuando se realizan en un mismo momento. Más adelante se verá una alternativa a esto, al tratar a las nociones de creencia o de conocimiento como conceptos que se pueden expresar con tres componentes lógicos en un mismo aparato formal.

La consistencia entre un conjunto de afirmaciones epistémicas formaliza de manera adecuada una idea implícita de conocimiento o creencia. El inconveniente de que sea un elemento fuera del lenguaje formal es que hace complicado diseñar un criterio para identificar la razón o evidencia que requiere el agente para aceptar o rechazar una afirmación epistémica. Al no contar con este elemento dentro del lenguaje, resulta indistinto hablar de una noción de creencia o conocimiento dentro del lenguaje, algo que reconoce implícitamente Hintikka al separar estas nociones en dos modalidades distintas, $K$ y $B$.

Para conocimiento se tienen las modalidades $K$, y su dual $P$, que representan que «el agente sabe que» y «el agente considera posible que», respectivamente. Para

creencia, Hintikka ofrece las modalidades *B* y *C*, que representan «el agente cree que» y «es compatible con las creencias del agente», respectivamente. Sin embargo, no hay una relación más específica entre creencia y conocimiento, o no queda clara la diferencia entre que el agente considere posible una afirmación (*P*) y que el agente crea una afirmación (*B*).

Por ejemplo, en la lógica epistémica de Hintikka se acepta el axioma **D**, que establece:

$$\Box\varphi \rightarrow \Diamond\varphi$$

que en términos de *K* sería:

$$K\varphi \rightarrow P\varphi$$

Es decir, lo que es necesario también es posible. Interpretado en un contexto epistémico, «*algo que sabemos, también lo consideramos posible*».

Esta idea de posibilidad es posible interpretarla como una noción débil de creencia. Que el agente conozca $\varphi$, implica que cree $\varphi$, porque sería extraño que no se creyera en algo que se afirma saber. Al menos no en un sentido de creencia, porque es posible formular afirmaciones del tipo «sé que ganó la elección, pero no lo puedo creer»[13], aquí es donde debemos tener cuidado con la idea de creencia que

[13] Decir «no lo puedo creer» se refiere aquí a un sentido figurado para expresar incredulidad o asombro, más que la imposibilidad de tener una creencia al respecto. Con esto no se pretende

se maneje. Es decir, en lógica epistémica, al menos de este modo planteada, no es posible establecer con la misma modalidad que:

$$K\varphi \rightarrow B\varphi$$

Y la razón puede ser que las condiciones para que sea conocimiento o creencia dependen de elementos externos al lenguaje lógico. Es decir, si pudiéramos especificar los conceptos de creencia y conocimiento, sería posible establecer este tipo de relación de manera más clara, sin necesidad de conceptos de uso restringido al lenguaje formal, como las modalidades *P* o *C*.

Con la idea de justificación integrada al aparato formal será posible definir esa noción especial de creencia o de conocimiento, para la que hay un elemento adicional, una especie de motivación o disposición del agente para generar la creencia o afirmación de conocimiento.

## 3.5. Justificación en lógica epistémica

Independientemente de la idea que se utilice de conocimiento, parece ser necesario contar con una noción de justificación. De acuerdo con Artemov en (S. Artemov,

disolver el problema de G. E. Moore que establece este caso de forma general como «*p*, pero no creo que *p*», simplemente estamos acotando el uso de creencia al diseño de este aparato formal para señalar el problema de distinguir entre dos nociones distintas de creencia.

2008), la justificación ha estado ausente en todos los intentos de formalizar nociones epistémicas. Sin embargo, la idea de justificación se puede ubicar, de forma implícita, en la formalización con lógica modal que utiliza Hintikka.

La formalización de conocimiento de Hintikka describe la noción de justificación de una afirmación epistémica (Hintikka, 1962, p. 20) relacionada con la relación de accesibilidad entre mundos posibles y una noción de consistencia para un conjunto de fórmulas.

> *[…] one can be justified in saying "I know" only if one's grounds are "conclusive" or "adequate" in some sense. I am not in a position to say "I know" unless my grounds for saying so are such that they give me the right to disregard any further evidence or information. […] Whoever says "I know that p" proposes to disregard the possibility that further information would lead him to deny that p although he could perhaps imagine (logically possible) experiences which could do just that. He is right if he is justified in doing so.*[14]

Hintikka en este pasaje se refiere a una noción contrafáctica para el conocimiento, es decir, se considera conocimiento si no es posible que el hecho que se afirma sea

[14] […] uno puede estar justificado en decir «yo sé» solo si se tienen razones «concluyentes» o «adecuadas» en algún sentido. No estoy en posición de decir «yo sé» a menos que mis razones para decirlo son tales que me permiten ignorar cualquier evidencia o información adicional. […] Quien diga «yo sé que *p*» lo hace con el propósito de descartar la posibilidad de que información adicional pueda llevarlo a negar *p* aunque pueda imaginar experiencias (lógicamente posibles) en las que hace precisamente eso. El sujeto está en lo correcto si tiene una justificación para ello.

refutado por alguna razón adicional. Aquí las condiciones contrafácticas se refieren a las alternativas o mundos posibles al alcance del agente cognitivo. Hintikka utiliza la relación de accesibilidad entre los mundos posibles como un recurso para evaluar las razones para aceptar una afirmación como conocimiento. Hintikka enuncia su noción de consistencia (Hintikka, 1962, p. 16) como sigue:

*(A.PKK *) If a set $\lambda$ of sentences is consistent and if* "$K_a p_1$" $\in \lambda$, "$K_a p_2$" $\in \lambda$, …, "$K_a p_k$" $\in \lambda$, "$P_a q$" $\in \lambda$, *then the set* {"$K_a p_1$", "$K_a p_2$", … , "$K_a p_k$", $q$} *is also consistent.*

Es decir, si en un conjunto de afirmaciones epistémicas $\lambda$, tenemos a un elemento que sostiene que el agente considera posible $q$, ($P_a q$), si es consistente ese conjunto de afirmaciones, quiere decir que no hay ninguna afirmación al alcance del agente que contradiga el conjunto. Por lo que, considerar $q$ como verdadera debe conservar la consistencia del mismo argumento.

Las condiciones que estipula Hintikka para establecer que la afirmación se conoce no son parte del lenguaje formal de la lógica epistémica (que nació a partir de este trabajo). Es decir, la justificación no tiene un elemento expresivo en la sintaxis del lenguaje, sino que se agrega con las condiciones de la relación de accesibilidad entre mundos. Las propiedades que debe cumplir la relación de accesibilidad entre mundos posibles son elementos «no lógicos», en cuanto a que no forman parte de las reglas sintácticas del lenguaje de la lógica epistémica. La justificación para afirmaciones de conocimiento en el aparato de Hintikka es un elemento externo al lenguaje, que se refiere a la consistencia de la afirmación epistémica respecto a los mundos accesibles para el agente.

El aparato formal puede no contener la idea explícita de justificación, pero Hintikka también considera que tiene un papel importante en la noción de conocimiento. Es decir, para Hintikka también es posible hablar de una noción tripartita para el conocimiento, lo que de algún modo sugiere que, al igual que Gettier, considera al conocimiento como algo que puede verse como una creencia verdadera justificada (definición necesaria). Sin embargo, no hay una relación directa entre creencia y conocimiento a menos que se relacionen modalidades distintas (*K* y *B*, por ejemplo), lo que sugiere que no es suficiente la definición de conocimiento como creencia verdadera justificada.

## 3.6. Conocimiento como una noción tripartita

Por otro lado, Artemov y otros (S. Artemov, 2008; S. N. Artemov, 1998; S. N. Artemov & Nogina, 2005) argumentan que la lógica epistémica y otras formalizaciones de conocimiento derivadas, no atienden la noción de justificación de la definición tripartita del conocimiento, por no incluir en el aparato formal una noción clara de justificación. Artemov tiene razón en señalar que en las formalizaciones anteriores no hay una expresión explícita de la justificación para afirmaciones epistémicas. Sin embargo, la idea de justificación de forma implícita está presente en la lógica epistémica (y modal, en general), algo que se hace evidente con los teoremas de elucidación que hacen explícitas fórmulas modales encontrando la forma del polinomio de justificación.

Utilizar una formalización como la lógica de justificación para afirmaciones de conocimiento, permite evitar los problemas de definir conocimiento como una *creencia verdadera justificada*, a favor de que, a través de un aparato capaz de representar

creencias, una noción de verdad y un modo de expresar la justificación de esas creencias; y la interpretación de las afirmaciones del lenguaje como conocimiento depende del contexto donde se aplique.

**Formalización del conocimiento**

Una formalización es muy útil para enfocar la atención en la información relevante. Por ejemplo, podemos pensar en el numeral «2» como una formalización de lo que sea que signifique el número dos e independientemente de la naturaleza de número, en general. A través de una formalización es posible esclarecer conceptos que de otra manera no sean tan sencillos de asimilar. Utilizar «2» en lugar de «dos» y una notación posicional, tuvo un papel fundamental en el desarrollo del álgebra. En Babilonia se utilizaba una notación posicional de números y fue ahí donde se desarrolló el álgebra y la idea de algoritmo. La información abstracta que contiene una formalización simbólica puede ayudar a ver cosas más complejas que se escapan cuando se trabaja con la información en crudo.

Definir el conocimiento como una creencia verdadera justificada permite descomponer un concepto problemático, como el conocimiento, en otros más simples. Esta definición ha demostrado cierta resistencia a ser modificada, y es que se puede rastrear incluso hasta los diálogos de Platón en Teetetos. Esta definición se ha mantenido incluso hasta nuestros días, sin olvidar el fuerte desafío de Gettier en 1963, señalando que es demasiado amplia y conduce a aceptar como conocimiento afirmaciones que no lo son.

Este trabajo no pretende profundizar sobre el debate de si el conocimiento es o no una creencia verdadera justificada. La idea es que, sea lo que sea el conocimiento, parece que puede descomponerse en tres componentes significativas.

- Algo que expresa una «creencia» (afirmación, proposición, pensamiento, opinión, etc.),
- algo que sirve para verificarla (validar, confirmar esa afirmación, compararla con lo que consideramos «verdadero», etc.),
- y algo que sirva como razón para formularla (explicación, «justificación», evidencia, intensión para esa afirmación, etc.), con independencia de si es o no verdadera.

En estos términos generales es que debemos ver al «conocimiento» del que se habla en lógica de justificación, con el propósito de ofrecer una formalización de esa noción. Si una formalización captura el carácter tripartito, será capaz de analizar ese tipo de afirmaciones dentro del mismo lenguaje, algo que la lógica modal epistémica no permite.

En epistemología, un problema fundamental es la naturaleza de las creencias de un sujeto cognitivo, sobre la verdad de esa creencia, y claro, la razón de la verdad de la creencia. Una formalización permite generalizar esa información en un modelo abstracto para poder operar sobre esos elementos en un aparto simbólico. En lógica y en computación esta formalización se puede utilizar como un esquema para diseñar mecanismos abstractos para representar conocimiento, tal como se hace en un lenguaje de programación.

Analicemos la fórmula $t{:}F$, que Artemov sugiere que se puede leer como:

> *El agente acepta la pieza de evidencia $t$ como justificación para $F$.*

O más brevemente:

> *$t$ es una justificación de $F$*

Si lo leemos en esos términos generales, no es necesario que de entrada demos una caracterización de lo que significa «aceptar una pieza de evidencia», o incluso la noción de justificación epistémica. El lenguaje permite expresar este tipo de afirmaciones sin tener que analizar el carácter de cada componente.

En lógica de justificación, lo que expresan los términos de justificación es una interpretación epistémica de la noción de demostración matemática, porque este lenguaje proviene de **LP** inicialmente. Una demostración es una razón para aceptar una afirmación codificada en el lenguaje, y si esa demostración convence o es una «buena» demostración, será verdadera.

En **LP**, las demostraciones son elementos sintácticos del lenguaje. Y las demostraciones interpretadas epistémicamente se pueden ver como razones para aceptar una afirmación en este lenguaje. Los polinomios de justificación contienen información que se puede utilizar para reconstruir la fórmula que están justificando; por ejemplo, si tenemos la fórmula justificada $(u \cdot v){:}F$, podemos reconstruir su derivación a partir de la justificación. En particular, con esa justificación sabemos que vino de una instancia del axioma de Aplicación, y en dos pasos anteriores tuvimos;

- $u{:}\,(\varphi \to F)$,
- $v{:}\,\varphi$;

para alguna fórmula $\varphi$ y los términos de justificación $u$ y $v$.

La idea de la especificación constante ($\mathcal{CS}$) juega aquí un rol muy importante, porque permite reconstruir una fórmula justificada a partir de esos supuestos, pero también puede efectuarse la reconstrucción a partir de un agente totalmente escéptico (si el conjunto de especificación constante es vacío). Esta reconstrucción estará acotada por los elementos de la especificación constante, por lo que, si la especificación constante es finita, tenemos razones para concluir que la reconstrucción tendrá un número finito de pasos, algo muy útil para poder medir la complejidad de la derivación de una fórmula.

Los términos de justificación son elementos abstractos ligados a una fórmula, abstractos en el sentido de que no tenemos que analizar su naturaleza más allá de la relación que tienen con la fórmula que justifican. De esta manera, podemos interpretar un término de justificación como percepción, evidencia empírica, hipótesis científica, demostración matemática, etc.

## 3.7. Aplicaciones

### Formalización de situaciones epistémicas

#### *Granero rojo*

En el capítulo anterior, vimos que la lógica de justificación permite operar con diferentes razones para un mismo hecho, y si aceptamos que se puede hacer un análisis externo en el que podemos juzgar si una afirmación es conocimiento o mera creencia, también podemos identificar las diferencias que tienen esas afirmaciones. Partiendo del ejemplo de los graneros rojos, tenemos:

$u{:}\, B$, $u$ es una razón para concluir que estoy viendo un granero ($B$).

Donde $u$ puede verse como mi percepción para identificar graneros con mis sentidos. Pero como vimos entonces, la percepción de poder identificar un granero puede no ser suficiente para concluir que es efectivamente un granero, o que «sabemos» que es un granero. Hace falta algo más para poder identificar una razón adecuada y concluir que lo que perciben mis sentidos es un granero.

Como vimos en ese ejemplo, cuando partimos de que estamos viendo un granero rojo $v{:}\, (B \wedge R)$, derivamos de ahí una razón para concluir que efectivamente es un granero, es decir $(a \cdot v){:}\, B$, donde $a$ es un término de justificación constante que obtuvimos de la tautología $(B \wedge R) \rightarrow B$. Respecto al papel de la justificación en este ejemplo, lo que está pasando es que, a través de un tratamiento simbólico de las razones, es posible llegar a la conclusión que se trata de un granero, lo que no

significa que hayamos solucionado el problema que señala Gettier, Goldman, Russell, etc. Con esta distinción es posible diagnosticar de manera más efectiva que tenemos dos maneras de «justificar» una misma afirmación. En este caso, una de ellas es incorrecta, porque estaríamos apelando a nuestra percepción de «granero» sin tomar en cuenta el color, lo que conduce a la posibilidad de que no sea un granero real, a diferencia de los graneros rojos que sí son reales en el universo de este ejemplo.

A continuación se presentan las diferencias entre las dos justificaciones para el hecho «frente a mí hay un granero».

- $u : B$. Si nuestros sentidos acerca de poder identificar graneros no están equivocados, estamos viendo un granero. Esta alternativa conduce a que, debido a que puede haber graneros falsos, podríamos no estar viendo un granero.

  Por lo tanto, esta afirmación no alcanza para caracterizar esta afirmación como conocimiento. Hay una situación en la que es posible que no sea un granero (por haber graneros falsos).

- $(a \cdot v) : B$. Si partimos de que nuestros sentidos están percibiendo un granero rojo, en particular estamos viendo un granero. Si consideramos este camino, y de acuerdo con las condiciones del ejemplo, tendríamos de hecho conocimiento.

Como podemos ver, las condiciones para aceptar la segunda opción son mucho más restrictivas que la primera, y de acuerdo con las condiciones del ejemplo, si

aceptamos que se puede hacer un análisis externo de este tipo de afirmaciones, $(a \cdot v) : B$ se puede considerar como conocimiento, mientras que $u : B$ falla, porque $u$ no es justificación suficiente para concluir que estamos viendo un granero, pero $(a \cdot v)$ sí.

El análisis de situaciones en las que las creencias del agente no permiten asumir si constituyen o no conocimiento requiere poder distinguir entre diferentes razones para un mismo hecho dentro del mismo aparato que opera sobre afirmaciones epistémicas y doxásticas. Esto se puede lograr diseñando sistemas multimodales, por ejemplo, que puedan manipular las creencias y conocimiento usando distintas modalidades. Sin embargo, el uso de aparatos multimodales incrementa la complejidad de las operaciones y en algunos casos incluso se sacrifica la decibilidad, una propiedad especialmente importante si se planea utilizar este tipo de razonamiento en entornos computacionales.

La lógica de justificación permite este análisis (distinguir entre creencias y conocimiento) y es por eso que representa una manera de responder a los casos presentados por Gettier y otros.

La lógica de justificación se puede utilizar para analizar casos en los que no podemos asumir de antemano si las afirmaciones son o no conocimiento, y para analizar propiedades específicas de conocimiento o creencia. Propiedades como la cerradura bajo el operador de conocimiento serán de especial atención en este trabajo.

### *Roca en forma de oveja*

Similar al caso anterior, analizaremos ahora un ejemplo que utiliza Pritchard (Pritchard, 2012) para argumentar en contra de lo que se conoce como suerte epistémica cuando se caracteriza una afirmación como conocimiento. Nos concentraremos en la formalización de este argumento para resaltar la utilidad de la lógica de justificación cuando se usa en estos casos para diagnosticar el carácter de una creencia. La interpretación de la «justificación» en este caso puede ser la razón que tiene el agente, además de la suerte, de sostener que una creencia es verdadera.

El ejemplo es el siguiente:

> *SHEEP: Roddy, in good epistemic conditions – in good light, at close range, and so on – sees what he takes to be a sheep, and so forms the belief that there is a sheep in the field. While this belief is true, in that there is a sheep in the field, Roddy is not looking at a sheep but rather a sheep-shaped object (such as a hairy dog). The genuine sheep is hidden from view behind the sheep-shaped object.*[15]

[15] OVEJA: Roddy, en buenas condiciones epistémicas – con buena luz, suficientemente cerca, etc. – visualiza lo que considera que es una oveja, por lo que forma la creencia de que es una oveja en el campo. Aunque su creencia es verdadera, porque hay una oveja en el campo, Roddy no está mirando a la oveja, sino a un objeto con forma de oveja (como un perro lanudo). La oveja genuina está oculta de su vista detrás del objeto con forma de oveja.

Roddy usa sus facultades de percepción y forma una creencia verdadera de que hay una oveja frente a él en el campo. Sin que Roddy sepa, el objeto que está mirando en el campo no es una oveja, sino una roca con forma de oveja que está ocultando a una oveja detrás.

- $S$, «frente a mí hay una oveja»
- $R$, «detrás de la roca hay una oveja real»
- $p{:}\, S$, $p$ es razón para creer que «frente a mí hay una oveja»
- $(R \wedge S)$, «frente a mí hay una oveja» y «detrás de la roca hay una oveja real»
- $q{:}\, (R \wedge S)$, $q$ es razón para creer que $(R \wedge S)$

La creencia $(R \wedge S)$ es pensable para un agente como Robby. Si Robby articula esta creencia, tiene manera de construir una justificación adecuada de que está viendo una oveja. La formalización de este ejemplo en lógica de justificación sería muy similar al granero rojo.

1. $p{:}\, S$, $p$ es razón para creer que «frente a mí hay una oveja»
2. $q{:}\, (R \wedge S)$, $q$ es razón para creer que $(R \wedge S)$
3. $a{:}\, (R \wedge S \rightarrow S)$ , $a$ es una justificación constante para la tautología $(R \wedge S \rightarrow S)$
4. $a{:}\, (R \wedge S \rightarrow S) \rightarrow q{:}\, (R \wedge S) \rightarrow (a \cdot q){:}\, S$, instancia del axioma de Aplicación

5. $q{:}\,(R \wedge S) \rightarrow (a \cdot q){:}\,S$, MP con (3) y (4)

6. $(a \cdot q){:}\,S$, MP con (2) y (5)

Estos ejemplos resaltan los problemas que podemos tener para diferenciar una creencia del conocimiento. Claramente el ejemplo busca que el lector sepa que «frente a mí hay una oveja» es problemático y no sea posible decidir si es conocimiento o solo una creencia verdadera.

Con lógica de justificación podemos distinguir entre $p{:}\,S$ y $(a \cdot q){:}\,S$, donde la primera es el resultado de solo usar la percepción del sujeto y la segunda es conocimiento en virtud de que Robby tenga la ocurrencia de imaginar que la forma que está viendo podría ser una roca exactamente igual a la oveja que está detrás. Esto segundo puede parecer como una respuesta *ad hoc,* pero el ejemplo está formulado en los mismos términos.

Por supuesto que se puede argumentar en contra del golpe de suerte de Robby al lanzar esta creencia tan arbitraria (imaginar que lo que está viendo es una roca con la forma de la oveja que está detrás). Sin embargo, si no se permite al agente cognitivo este tipo de afirmaciones aparentemente al azar, también estaríamos eliminando la posibilidad de aciertos por casualidad y la misma idea de creatividad de los agentes que estamos modelando, que al menos en principio, son humanos. Y es que no solo estamos hablando de suerte con esto, incluso podemos pensar en una hipótesis científica, que si bien se formula con información antecedente, el científico no está absolutamente seguro de su hipótesis, porque si lo estuviera, ¿para qué hacer el experimento en primer lugar?

## Cerradura de conocimiento con justificaciones

La cerradura del conocimiento es una propiedad epistémica que establece que si un agente *S* sabe una proposición *P*, y además sabe que *P* implica lógicamente a *Q*, entonces *S* debe saber también *Q*, nada más es necesario. Si el conocimiento es cerrado bajo implicación de conocimiento[16], un agente sabe todas las implicaciones de lo que sabe.

En esta sección se formalizarán algunos aspectos de este argumento con lógica de justificación para ayudar a diagnosticar el problema y ofrecer nuevos elementos de análisis. La lógica de justificación no debe verse como una fórmula mágica para resolver debates filosóficos, pero ayuda a esclarecer elementos que no es tan sencillo visualizar sin una herramienta como esta.

Dretske en (Dretske, 2005) sostiene que el siguiente argumento no es válido:

- *S* sabe *P*
- *S* sabe que *P* implica *Q*
- por lo tanto, *S* sabe *Q*

[16] Dretske distingue entre *modus ponens* y la implicación de conocimiento. *Modus Ponens* establece que si *P* y $P \rightarrow Q$ son verdaderas, también debe ser verdadera *Q*. La cerradura de conocimiento es más restrictiva, ya que si el agente *S* sabe *P*, y también sabe $P \rightarrow Q$, por MP, *Q* también es verdad, pero la cerradura además permite concluir que *S* sabe *Q*.

Que no sea válido no significa que sea falso siempre, sino que no se cumple en el caso general. Presentar un ejemplo en el que se mantenga la cerradura no es prueba de que el conocimiento tenga la propiedad de ser cerrado. Dretske prepara su argumento en contra de la cerradura con la propiedad de la transmisión de razones para creencias, una versión más débil. Esto con el fin de señalar que una razón o pieza de evidencia para una afirmación no tiene por qué funcionar para las implicaciones de lo que ya se conoce. El argumento de la transferencia de razones es el siguiente:

- $r$ es una razón para que $S$ crea que $P$
- $S$ sabe que $P$ implica lógicamente a $Q$
- $r$ es una razón para que $S$ crea que $Q$

El argumento se puede formalizar en lógica de justificación como:

$$u\colon (P \to Q), r\colon P \vdash r\colon Q$$

donde $u$ es la justificación de $S$ para $(P \to Q)$, y por el teorema de elucidación, sabemos que se puede obtener el polinomio $u$ a partir de la afirmación de conocimiento «$S$ sabe que P implica lógicamente a $Q$».

Dretske sostiene que este argumento *no es válido*, y esto en lógica de justificación es inmediato, porque basta dar un modelo en el que se cumpla $u\colon (P \to Q)$ y $r\colon P$ pero no $r\colon Q$. Del mismo modo que se discutió en el capítulo § 2 acerca de que la lógica de justificación no es una lógica modal normal, al operar con justificaciones

explícitas, se puede construir el modelo $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{V}, \mathcal{E} \rangle$, que sirve de contra-ejemplo para el argumento, donde:

$\mathcal{W} = \{1\}$; $\mathcal{R} = \{\}$; $\mathcal{V}(1) = \{P, Q\}$; $\mathcal{E}(u, P \to Q) = \{1\}$, $\mathcal{E}(r, P) = \{1\}$,

Por lo tanto, en $\mathcal{M}$ se cumple:

$\mathcal{M}, 1 \vDash u{:}\,(P \to Q)$

$\mathcal{M}, 1 \vDash r{:}\,P$

Pero esto, por la la propiedad *E1* (2.3.2) de $\mathcal{E}$, solo garantiza

$\mathcal{M}, 1 \vDash (u \cdot r){:}\,Q$

Con lógica de justificación es posible distinguir entre la afirmación $(u \cdot r){:}\,Q$ y $r{:}\,Q$, algo que una formalización modal implícita no es posible. Y basta con que $1 \notin \mathcal{E}(r, Q)$ para que el argumento no se cumpla, es decir, con lógica de justificación es posible expresar la situación en la que, teniendo evidencia para $P$, y aceptando como conocimiento $(P \to Q)$ la evidencia para $P$ no se transmite a $Q$.

■

Utilizar un aparato formal para expresar estas relaciones permite analizar propiedades antes no expuestas, como las propiedades que pueden cumplir las operaciones entre términos de justificación o el papel de la función de evidencia en la aceptación de una fórmula epistémica. Por supuesto, esto no resuelve el problema de determinar si la percepción es suficiente para obtener conocimiento, o si los casos

Gettier constituyen creencias verdaderas con una justificación, pero no conocimiento. El aparato formal simplemente expone los elementos de forma abstracta para que el análisis pueda ser más granular. Tampoco es necesario comprometerse con la idea de que el conocimiento es una creencia verdadera justificada, pero es útil ver esa noción como una entidad tripartita que se puede analizar por separado.

# 4. Omnisciencia lógica

## 4.1. Introducción

Entenderemos la omnisciencia lógica como la característica de un agente epistémico (humano o computacional) de conocer todas las consecuencias lógicas de sus afirmaciones. Normalmente se le asocia con el axioma de distribución, conocido también como **K**, para la lógica epistémica:

$$K(\varphi \rightarrow \psi) \rightarrow K\varphi \rightarrow K\psi$$

Que se lee como, «si el agente sabe $(\varphi \rightarrow \psi)$ y también sabe $\varphi$, está en posición de saber $\psi$».

La lógica epistémica modela el conocimiento a través de la idea de necesidad y posibilidad que se expresa en la lógica modal. El axioma de distribución se acepta en las lógicas modales «normales», que son casi todas. Se puede argumentar (Stalnaker, 1991) que este modo de aceptar el axioma de distribución es una especie de idealización para obtener un modelo del conocimiento. Idealización en el sentido de que el principio de cerradura de conocimiento es obviamente falso, así como es falso que exista un plano infinito o un gas ideal, pero funciona como una heurística para modelar las actitudes epistémicas de un agente.

Sin embargo, también hay intentos de ajustar los sistemas lógicos para que no presenten las características de la omnisciencia lógica. Estos sistemas añaden axiomas y se modifican las estructuras lógicas que determinan el modo de inferencia para evitar la omnisciencia. Sim recopiló (Sim, 1997) varios sistemas lógicos que evitan

la omnisciencia lógica, uno de ellos es la lógica de creencias implícitas y explícitas de Levesque. Por su manera de caracterizar las propiedades que debe cumplir una lógica respecto a la omnisciencia lógica, se hará una breve descripción de este aparato.

**Lógica de creencias implícitas y explícitas**

Levesque (Levesque, 1984) propone una lógica que separa la idea de creencias implícitas y explícitas, donde identifica las siguientes como desventajas de aceptar la omnisciencia lógica en el aparato formal con el que se trabaje:

1. Toda afirmación válida debe ser considerada creencia[17].
2. Si dos afirmaciones son lógicamente equivalentes, si una es considerada creencia, la otra también.
3. Si una afirmación y su negación son consideradas creencias, cualquier otra afirmación también debe serlo.

La idea principal de la lógica que propone Levesque es que las creencias se pueden separar en implícitas y explícitas. Una creencia explícita es la que está al alcance del agente, mientras que una creencia implícita es una consecuencia lógica de una creencia explícita. La diferencia principal con la lógica epistémica tradicional es

[17] Levesque formaliza nociones de creencia y no de conocimiento, aunque su análisis en lo general sobre creencias implícitas y explícitas se puede extender también a conocimiento.

que utiliza la idea de *situaciones*, que sustituyen a los mundos posibles para representar las creencias de los agentes. La semántica se separa en dos nociones independientes de verdad y falsedad. El modelo para esta lógica es la tupla $\mathcal{M}_L = \langle S, \Sigma, T, F \rangle$, donde $S$ es un el conjunto de situaciones; $\Sigma$ es un subconjunto de $S$; $T$ y $F$ son funciones de mapeo de átomos proposicionales *Atom*, a subconjuntos de $S$, tal que $p \in Atom$, $T(p)$ representa las situaciones verdaderas para $p$, y $F(p)$ las situaciones falsas. Una situación es incoherente si está en el conjunto $T(p) \cap F(p)$, y es incompleta si no está en $T(p) \cup F(p)$. Un mundo posible $w$ es una situación completa y coherente.

El lenguaje de la lógica de Levesque se construye con las conectivas proposicionales estándar $\wedge$, $\vee$, $\neg$[18], y dos operadores epistémicos, $B$ para creencias explícitas y $L$ para creencias implícitas. Solo fórmulas proposicionales (sin $B$ o $L$) pueden estar en el ámbito de los operadores.

Para hacer la idea de situación compatible con los mundos posibles, se establece la relación de un mundo posible $w$ con una situación $s$, $\mathcal{W}(s)$,

$\mathcal{W}(s) = \{ s' \in S \mid$ *para todo* $p \in Atom$,

a) $s'$ es elemento de $T(p)$ o $F(p)$ pero no ambos,

[18] El operador de implicación $\rightarrow$, se define en la lógica de Levesque como $(\neg\phi \vee \psi)$. Esto lo hace con el fin de utilizar solo disyunción y conjunción en su semántica y aplicar equivalencias de De Morgan.

b) si $s$ es elemento de $T(p)$, entonces también $s'$ lo es,

c) si $s$ es elemento de $F(p)$, entonces también $s'$ lo es. }

Por lo tanto, $\mathcal{W}(\Sigma)$ representa el conjunto de mundos posibles involucrados.

Requiere de dos relaciones de consecuencia lógica, $\models_F$ y $\models_T$. Para una situación $s$ y fórmulas del lenguaje $\alpha$ y $\beta$:

1. $s \models_T p$, ssi $s \in T(p)$, con $p \in Atom$

   $s \models_F p$, ssi $s \in F(p)$, con $p \in Atom$

2. $s \models_T (\alpha \vee \beta)$, ssi $s \models_T \alpha$, o bien $s \models_T \beta$

   $s \models_F (\alpha \vee \beta)$, ssi $s \models_F \alpha$, y $s \models_F \beta$

3. $s \models_T (\alpha \wedge \beta)$, ssi $s \models_T \alpha$, y $s \models_T \beta$

   $s \models_F (\alpha \wedge \beta)$, ssi $s \models_F \alpha$, o bien $s \models_F \beta$

4. $s \models_T \neg\alpha$, ssi $s \models_F \alpha$

   $s \models_F \neg\alpha$, ssi $s \models_T \alpha$

5. $s \models_T B\alpha$, ssi para toda $s'$ en $\Sigma$, $s' \models_T \alpha$

   $s \models_F B\alpha$, ssi $s \not\models_T B\alpha$

6. $s \models_T L\alpha$, ssi para toda $s'$ en $\mathcal{W}(\Sigma)$, $s' \models_T \alpha$

   $s \models_F \alpha$, ssi $s \not\models_T L\alpha$

Los operadores $B$ y $L$ están relacionados por la afirmación válida[19]

$$(B\alpha \rightarrow L\alpha)$$

que establece que una toda creencia explícita también es una creencia implícita.

Para el operador de creencias implícitas $L$, las tautologías proposicionales son válidas. Respecto a las propiedades omniscientes de esta lógica, el operador de creencias implícitas cumple con el axioma de distribución, tal que:

$$\models L(\alpha \rightarrow \beta) \rightarrow L\alpha \rightarrow L\beta$$

El operador de creencias explícitas $B$, por otro lado, está diseñado por Levesque para evitar los que considera conjuntos de afirmaciones que deben ser satisfactibles en un sistema no omnisciente.

1. $\{B(\varphi \rightarrow \psi), B\varphi, \neg B\psi\}$[20], las creencias explícitas no son cerradas bajo implicación lógica.
2. $\{\neg B(\varphi \vee \neg\varphi)\}$, una afirmación válida (una tautología), no necesita ser considerada creencia.
3. $\{B\varphi, \neg B(\varphi \wedge (\psi \vee \neg\psi))\}$, un equivalente lógico a una creencia no necesita ser considerado creencia.

[19] Una fórmula $\alpha$ en la lógica de Levesque es válida ($\models \alpha$), si se cumple para todo modelo $\mathcal{M}_L = \langle S, \Sigma, T, F\rangle$.

[20] Al satisfacer este conjunto de afirmaciones, el operador $B$ de Levesque no cumple con el axioma de distribución.

4. $\{B\varphi, B\neg\varphi, \neg B\psi\}$, las creencias pueden ser inconsistentes sin considerar creencia cualquier otra creencia.

La lógica de Levesque captura características clave de un agente que puede formular creencias explícitas e implícitas sin presentar características omniscientes. Este enfoque, sin embargo, conduce a soluciones que se perciben *ah hoc* por la manera en que se manipulan los axiomas que se aceptan en los sistemas lógicos que se pretende que no sean omniscientes. También al diseñar aparatos multimodales que representen nociones de creencia distintas, se marca una línea entre una noción de creencia en el uso común de un agente y una noción «culta» restringida solo a una formulación simbólica deductiva.

**El enfoque de la lógica de justificación**

La lógica de justificación hace un análisis más detallado de las afirmaciones de conocimiento y permite obtener conclusiones que van más allá de la lógica epistémica convencional. Esto en términos formales se traduce en que la lógica de justificaciones puede expresar las mismas afirmaciones de conocimiento que las lógicas epistémicas normales, y además puede expresar la razón para esas afirmaciones.

La lógica de justificación, al ser una familia de lógicas que toman en cuenta el tratamiento de evidencia para afirmaciones, no requiere de hacer una distinción particular entre diferentes tipos de creencia porque una proposición epistémica en lógica de justificación puede interpretarse como creencia o conocimiento sin modificaciones en el aparato formal. Por lo tanto es posible formalizar este tipo de

situaciones con el mismo aparato, sin necesidad de integrar nuevas modalidades al lenguaje.

Es válido preguntar si un agente está en posibilidad de conocer $q$ sin ningún esfuerzo adicional partiendo de que está a su disposición conocer $p$ y $(p \rightarrow q)$, en particular cuando el objetivo involucra la automatización o mecanización de estos procesos de inferencia. La lógica de justificación atiende este enfoque al tratar el problema utilizando la teoría de complejidad computacional (de cuantificación de recursos) en lugar de un problema de etiquetación de tipos o clases de creencia para los agentes (que corresponde con asignar modalidades distintas a cada caso).

A continuación, se revisará la respuesta a la omnisciencia lógica desde el punto de vista de la lógica de justificación. Desde esta perspectiva, la omnisciencia se trata como un problema simbólico, porque la lógica de justificación permite razonar directamente con términos de evidencia y un sistema lógico no será omnisciente si es posible encontrar una secuencia polinomial de pasos para demostrar una afirmación del tipo «el agente sabe $F$».

Para poder hacer este análisis, será necesario revisar algunos conceptos de complejidad computacional en el contexto de demostraciones lógicas, siguiendo el desarrollo de Cook-Reckhow en (Cook & Reckhow, 1974) y (Pudlák, 1998). Para especificar los pasos de la derivación, se revisará el sistema de tableauxs semánticos (D'Agostino, Gabbay, Hähnle, & Posegga, 2013), y finalmente con el método de verificación diseñado por Artemov y Kuznets (S. Artemov & Kuznets, 2006) de que **LP** no es omnisciente y puede decidirse en tiempo polinomial con un polinomio en la longitud de la fórmula que se está demostrando.

## 4.2. Complejidad en demostraciones

El estudio cuantitativo de las demostraciones se refiere no solo a decidir si un teorema es verdadero o falso, sino si su demostración es tratable (que pueda escribirse en papel o que su verificación pueda realizarse por una computadora).

Los sistemas de inferencia que tengan la propiedad del *corte* permiten construir demostraciones más cortas, por la capacidad de utilizar «lemas», es decir, demostraciones con resultados intermedios. Si el sistema de demostraciones admite la eliminación del corte (*cf. cut elimination*), la longitud de la demostración puede aumentar exponencialmente o peor (las demostraciones en aritmética crecen superexponencialmente), por lo que la propiedad del corte se convierte en un tema fundamental a la hora de obtener la longitud de una demostración o medir el tiempo de ejecución de un algoritmo para realizarla computacionalmente.

El estudio de complejidad en demostraciones parte de la idea de que los problemas computacionalmente difíciles, como tomar partido de la pregunta **P = NP**, son problemas esencialmente lógicos y no combinatorios (Pudlák, 1998). Si este es el caso, la teoría de demostraciones, y en particular obtener la longitud de una demostración, debe jugar un papel importante en la solución de este tipo de preguntas.

### Demostraciones y medidas de complejidad

Una demostración debe ser computable en tiempo polinomial, de acuerdo a un polinomio en la longitud de la fórmula que se está demostrando (Pudlák, 1998).

Esto es asumiendo que las demostraciones y las fórmulas están codificadas en cadenas de un alfabeto finito y que identifiquemos una computación eficiente como una computación en tiempo polinomial.

Arora en (Arora & Barak, 2009, p. 25) sostiene que la clase de complejidad **P** se considera como una buena definición de cómputo efectivo en el contexto de justificaciones teóricas sobre la existencia de algoritmos subexponenciales para problemas, y también desde el punto de vista del programador, para poder determinar hasta qué punto es posible optimizar la solución de un problema. De este modo se hace la conexión entre la teoría de demostraciones y la teoría de complejidad computacional.

Un mecanismo de decisión que determine si una fórmula es correcta o no, de acuerdo con las reglas de inferencia de un sistema formal, es un sistema de demostraciones en términos muy generales que permite construir un procedimiento de decisión no determinista para el conjunto de tautologías (que lo hace parte de la clase de complejidad **coNP**), que también se pueden ver como los teoremas de una teoría o sistema formal en particular.

En teoría de complejidad es fundamental adoptar una medida de tiempo para la ejecución de operaciones, y determinar cotas superiores e inferiores en esos términos. En un sistema de demostraciones, el tamaño de la demostración está determinado por el número de pasos utilizando las reglas de inferencia del sistema.

La cota superior sería el número de posibilidades, elegidas no determinísticamente, de derivar una fórmula dentro del procedimiento de demostraciones que se tenga. Una cota inferior sería un procedimiento «eficiente» para encontrar la

demostración dentro del sistema formal de inferencia. Por ejemplo, sea *TAUT* el conjunto de tautologías en lógica proposicional (***L***) de fórmulas codificadas en un alfabeto, con un conjunto completo de conectivas lógicas (p. ej. conjunción $\wedge$ y negación $\neg$). Un sistema de demostraciones $\mathcal{D}$ es una relación binaria $P(x, y)$, que es una función computable en tiempo polinomial, donde:

$$TAUT \equiv \{\varphi \in \boldsymbol{L} \mid \exists y P(\varphi, y)\}$$

donde $\varphi$ es una fórmula proposicional, y $y$ es una demostración para la fórmula $\varphi$ en el sistema de demostraciones $\mathcal{D}$ con la función binaria (que expresa la demostración) $P$.

El conjunto *TAUT* pertenece a la clase de complejidad **coNP**-completo (Arora & Barak, 2009, p. 56), por lo que se cumple el siguiente teorema respecto a los sistemas de demostración.

*Teorema*. Cook-Reckhow (1979). Existe un sistema de demostraciones para la lógica proposicional en el que todas las tautologías (teoremas) tienen una demostración de longitud polinomial de acuerdo a la longitud de la fórmula, si y solo si **NP = coNP**.

■

En el caso particular de la lógica, un sistema de demostraciones está dado por una lista finita de reglas de deducción. El elemento básico de una demostración es un paso de derivación o una línea en la demostración, que puede ser una fórmula, un

conjunto de fórmulas, una secuencia de fórmulas o un secuente (que está formado por dos secuencias de fórmulas).

Una *demostración* en estos términos es una secuencia de fórmulas o un árbol de los pasos de una demostración, tal que cada paso es un axioma de la teoría o se sigue de los pasos previos a través de las reglas de deducción.

La medida más importante en la teoría de complejidad en demostraciones es el *tamaño* de la demostración. La demostración se forma con un alfabeto finito de símbolos y una manera de codificar las secuencias de la demostración también en un alfabeto finito. El tamaño de una demostración es la longitud en el número de símbolos en la codificación. Otra medida es el *número de líneas* de una demostración, es decir, el número de pasos. Trivialmente, el número de líneas es, a lo más, igual al tamaño de la demostración, pero ésta puede consistir de fórmulas muy largas, por lo que vale la pena conservar ambas medidas. Otra razón para conservar ambas medidas es que las demostraciones de un solo paso (por ejemplo las que coinciden con tautologías) tienen el mismo número de pasos (uno) y no se podría distinguir solamente con el número de pasos si fuera la única medida.

Haciendo una analogía con la teoría de la complejidad, el tamaño de la demostración corresponde con la medida de tiempo en complejidad computacional, pero por sí sola, la medida de número de líneas de demostración no corresponde con la

de espacio. El espacio en complejidad corresponde con una medida maximal[21] de tiempo, pero en demostraciones no podemos hacer esa misma relación con la medida de número de líneas de demostración. Por ejemplo, cuando se presenta una demostración en el pizarrón o cuando una computadora verifica la validez de una demostración, no podemos ir línea por línea verificándola por separado, es necesario que se complete la demostración y debemos tener disponibles (por ejemplo, a la vista) las premisas necesarias para los pasos de derivación para continuar con el desarrollo de la demostración.

El tamaño de una fórmula $\varphi$ respecto a una demostración $d$ se denota por $|\varphi|$ *resp.* $|d|$. Sean $\mathcal{A}$ un sistema de demostraciones y $\varphi$ una fórmula demostrable en $\mathcal{A}$, se tiene que $\mathcal{A} \vdash^{n} \varphi$, si $\varphi$ tiene un tamaño de demostración menor o igual a $n$ pasos en $\mathcal{A}$.

## 4.3. Omnisciencia lógica como un problema de complejidad

El enfoque del problema desde la lógica de justificación está basado en definir una demostración dentro de un sistema lógico epistémico (una lógica que pueda lidiar con afirmaciones del tipo «el agente sabe *F*» en general), para determinar si tiene o no la característica de ser omnisciente. Esta prueba está basada en la teoría de

[21] Es decir, si tuviéramos tiempo infinito o una medida independiente del tiempo que se demore, ¿qué podría limitar la ejecución del procedimiento de decisión para el problema?

complejidad computacional y en particular en el enfoque de Cook-Reckhow sobre complejidad en demostraciones.

La omnisciencia lógica desde esta perspectiva se refiere al *costo computacional* de realizar inferencias en el sistema lógico epistémico del que se hable. De esta manera, la omnisciencia lógica se trata como un problema simbólico y se busca caracterizar la razón de las afirmaciones de conocimiento. La esencia de la omnisciencia lógica se encuentra en el carácter no constructivo de la lógica modal con la que se formaliza el conocimiento.

En un lenguaje modal, se atiende a las afirmaciones de conocimiento pero no a su origen, lo que se presta a aceptar afirmaciones de conocimiento sin una adecuada justificación, lo que lleva finalmente a afirmaciones que resulta problemático aceptar. Con estas consideraciones, a la omnisciencia lógica la definiremos de la siguiente manera (S. Artemov & Kuznets, 2006):

> *Un sistema epistémico $E$ no es lógicamente omnisciente si para cada afirmación de conocimiento válida $A$ del tipo "$F$ es conocimiento" existe una demostración de $F$ en $E$, y la complejidad de dicha demostración está acotada por un polinomio en la longitud de la afirmación $A$.*

Usando esta definición de omnisciencia se puede mostrar que las lógicas modales convencionales no pasan esta prueba, y por lo tanto son omniscientes. Lo que coincide con la intuición de que se está capturando la esencia del problema correctamente.

Esta caracterización además permite construir sistemas epistémicos que no sean lógicamente omniscientes. Esto significa que es posible tener sistemas epistémicos que puedan operar con más información acerca de la razón de «*F* es conocimiento», es decir, evidencia de dicha afirmación. Un sistema no omnisciente es la lógica de justificación **LP**, la versión explícita de **S4**.

## Verificación de Omnisciencia Lógica

En (S. Artemov & Kuznets, 2006) se introduce la *Verificación de Omnisciencia Lógica* (*cf. Logical Omniscience Test*), que caracteriza a los sistemas omniscientes. En ella se sigue la idea de medir la longitud de una demostración usando un sistema de demostraciones ***L***, que básicamente permite construir la prueba a partir de símbolos de un alfabeto $\Sigma$, y ve una «demostración» como una función $p\colon \Sigma^* \to \boldsymbol{L}$, que mapea cadenas de ese alfabeto en fórmulas válidas del sistema ***L***. También se considera la longitud de la demostración en el número de pasos con la función $\ell\colon \Sigma^* \to \mathbb{N}$, y la longitud de cada fórmula con el número de símbolos que las componen con $|\cdot|\colon Fm_L \to \mathbb{N}$.

4.3.1. *LOT. Verificación de Omnisciencia Lógica (cf. Logical Omniscience Test)* (S. Artemov & Kuznets, 2006). Sea ***L*** una teoría capaz de expresar afirmaciones de conocimiento de la forma «la fórmula *F* es conocimiento», un sistema de demostraciones *p*, una medida para el tamaño de las demostraciones $\ell$, y una medida para el tamaño individual de cada fórmula $|\cdot|$, si existe un polinomio *P*, tal que, para cada afirmación de conocimiento $\mathcal{A}$ en ***L*** de la forma «*F* es conocimiento», la fórmula *F* tiene una demostración $\mathcal{D} \in \Sigma^*$, tal que:

$$\ell(\mathcal{D}) \leq P(|\mathcal{A}|)$$

es decir, la demostración de la fórmula $F$ no puede ser mucho más grande que la propia afirmación.

Estas medidas dependerán del estilo de sistema de demostraciones que se utilice. Aquí se utilizará un sistema de Hilbert, que considera las siguientes medidas:

1. el número de pasos de la demostración,
2. el número de símbolos lógicos en la derivación,
3. el tamaño en bits de la derivación, es decir, la longitud de la cadena[22] respecto al alfabeto $\Sigma$.

Es importante tener en cuenta las diferencias de cada una de estas medidas. Es posible pensar un mecanismo que cuantifique las demostraciones solo por el número de fórmulas, es decir $|F| = \ell(F)$, pero esto haría que en todos los casos tuviéramos para el primer paso de la derivación $|F| = 1$, lo que no hace justicia a una derivación general. Con la medida en símbolos se pueden diferenciar estos casos.

Ya estamos en posición de caracterizar a la lógica modal como un sistema omnisciente, en particular se analizará el caso de **S4** y como medida el número de pasos

[22] Por la medida en bits de la derivación, podríamos considerar que $\Sigma$ es $\{0, 1\}$, aunque no es necesario hacer esa suposición. Basta con una manera de codificar de las cadenas respecto a cualquier alfabeto. Por ejemplo, con $\Sigma = \{0\}$, podríamos dar una codificación de modo que $'a' = '0'$, $'b' = '00'$, etc.

de la derivación, también se puede revisar el caso de la longitud de la demostración respecto a las conectivas y al tamaño en bits de las fórmulas en (S. Artemov & Kuznets, 2006).

*Teorema* (S. Artemov & Kuznets, 2006). Dado cualquier sistema de demostraciones *p* para **S4**, donde el tamaño de una demostración será el número de pasos en la derivación de esa fórmula. **S4** es lógicamente omnisciente, a menos que **PSPACE = NP**.

*Demostración*. Por contradicción, supongamos que **S4** no es lógicamente omnisciente, para cada afirmación de conocimiento *KF*, la fórmula *F* tiene una demostración polinomial en el sistema *p*, es decir, existe un polinomio *P* tal que, para cada derivación $\mathbf{S4} \vdash KF$ existe una demostración $\mathcal{D}_F$ de *F* que cumple la condición $\ell(\mathcal{D}_F) \leq P(|KF|)$.

Por la suposición de que **S4** no es lógicamente omnisciente, podemos construir un algoritmo en la clase **NP** para decidir una fórmula. Sea *G* una fórmula en **S4**, con lo que tenemos $\mathbf{S4} \vdash G$, y por el axioma **T**, tenemos también $\mathbf{S4} \vdash KG$, por lo que podemos determinar si la fórmula *G* es válida de manera no determinista si adivinamos la demostración de longitud polinomial en el sistema *p*, que también sería un polinomio en la longitud de la fórmula, $|KG|$.

Por otro lado, la derivación de una fórmula en **S4** es un problema **PSPACE**-completo. Debido a esto, y a la existencia de un algoritmo en **NP** para **S4**, permitiría

concluir que **PSPACE** $\subseteq$ **NP**, y por tanto la jerarquía polinomial colapsa en el segundo nivel, $\Sigma_1^P$.

■

Por otro lado, para demostrar que las lógicas de conocimiento explícito no son omniscientes bajo esta prueba, se utilizará la lógica de justificación **LP**.

Es necesario especificar la forma que tienen las afirmaciones de conocimiento, que a primera vista sería $t{:}F$, sin embargo, se debe considerar que, tanto $t$ como $F$ pueden tener términos de justificación constante. Esto se puede especificar si se considera como parte de la afirmación de conocimiento al conjunto de especificación constante $\mathcal{CS}$, que contiene las fórmulas que se consideran verdaderas (axiomas) con sus respectivos términos constantes.

Una mejor caracterización de las afirmaciones de conocimiento en **LP**, es:

$$\bigwedge \mathcal{CS} \rightarrow t{:}F$$

donde $\mathcal{CS}$ es el conjunto de especificación constante finito, y también se le puede considerar inyectivo, es decir, que un término constante justifique solo a un axioma. $\bigwedge \mathcal{CS}$ representa la conjunción de las fórmulas que pertenecen al conjunto $\mathcal{CS}$ y especificará todos los términos de justificación que aparezcan en $t$.

Como una derivación en **LP** usa un número finito de aplicaciones de la regla de internalización, cada derivación de $F$ puede convertirse en una derivación de $\mathbf{LP}_{\emptyset}$ de $\bigwedge \mathcal{CS} \rightarrow F$.

Aquí se considerará el número de pasos de la derivación como medida para las demostraciones, en (S. Artemov & Kuznets, 2006) también se analiza el caso del tamaño de las fórmulas y de las cadenas de cada paso.

Considerando el número de pasos de la derivación de una fórmula $F$, Roman Kuznets formuló un algoritmo en el que es posible derivar en **LP** una fórmula $F$ en un número polinomial de pasos respecto a la longitud de la fórmula $t{:}F$, sin considerar el conjunto de especificación constante.

El sistema de Hilbert para **LP** utiliza las siguientes reglas de inferencia, que coinciden con los axiomas de la lógica de justificación **LP**, que son los miembros de $\mathcal{CS}$.

| | | |
|---|---|---|
| $\dfrac{t{:}F}{!t{:}t{:}F}$ | $\dfrac{s{:}F}{(s+t){:}F}$ $\qquad$ $\dfrac{t{:}F}{(s+t){:}F}$ | $\dfrac{s{:}(F \to G) \quad t{:}F}{(s \cdot t){:}G}$ |
| Verificación | Suma | Aplicación |

*Tabla 1. Reglas de inferencia para **LP**.*

*Teorema* (S. Artemov & Kuznets, 2006). **LP** no es lógicamente omnisciente con respecto a un sistema de Hilbert y una medida de demostraciones con el número de fórmulas en ella.

*Demostración*. A continuación se mostrará que para cada afirmación de conocimiento $t{:}F$, existe una derivación en un sistema de Hilbert de $F$ que toma un número lineal de pasos. De hecho, a lo más $3|t| + 2$ pasos, donde $|t|$ es el número de símbolos en $t$.

De acuerdo con las reglas de derivación definidas para **LP**, tenemos que **LP** $\vdash t{:}F$, se puede derivar en a lo más $|t|$ pasos, porque cada regla incrementa el tamaño del polinomio de justificación de $F$, en al menos un símbolo.

Cada regla definida a partir de un axioma en **LP** es una instancia de la regla de internalización. Para derivar cada fórmula es suficiente aplicar la regla de *internalización* en **LP** para cada axioma y una o dos veces la regla Modus Ponens (*MP*) para obtener cada fórmula que se puede derivar por este sistema. Para la regla de aplicación (regla $\cdot$) se requiere una aplicación de *MP* para obtener una fórmula $(s \cdot t){:}F$, para la regla de verificación (regla !); se necesita una aplicación de *MP* para obtener una fórmula $!t{:}(t{:}F)$; finalmente, para obtener $(s + t){:}F$ es necesario aplicar dos veces *MP*, una para hacer la derivación en términos de $s$, y otra en términos de $t$. Con esto hemos descrito la manera de obtener los posibles polinomios de justificación. Resta derivar la fórmula $F$ a partir de $t{:}F$, para alguna $t$, para lo que necesitaremos utilizar el axioma de factibilidad, $t{:}F \rightarrow F$, y obtener así $F$ con una aplicación más de Modus Ponens. De esta forma queda descrito el proceso para obtener la afirmación de conocimiento $F$ en el lenguaje de *LP*, que toma a lo más $3|t| + 2$ pasos en el sistema de Hilbert para el lenguaje.

■

El objetivo de la prueba *LOT*, es tratar la omnisciencia lógica como un problema de complejidad computacional y la idea central es que el conocimiento puede verse como una manera de operar sobre evidencias de manera explícita a través de términos de justificación, en un aparato formal. Esta idea tiene que ver con la relación

que se puede establecer entre una demostración matemática y las afirmaciones epistémicas, que está reflejada en la lógica de justificación, al venir directamente de **LP**, que es una lógica de demostraciones.

## 4.4. Posturas alternas

Hay otras maneras de tratar la omnisciencia lógica. Halpern y Pucella en (Halpern & Pucella, 2007) hacen un análisis de alternativas para atender el problema de la omnisciencia lógica desde enfoques distintos: un tratamiento *sintáctico*, de percepción (*awareness*), *conocimiento algorítmico* y de *mundos imposibles*. Este análisis se hace con la idea de las posibles aplicaciones en el ámbito computacional de una respuesta satisfactoria al problema de la omnisciencia lógica. En el siguiente fragmento de ese artículo, se puede ver la relación de los sistemas omniscientes en el contexto computacional.

> *While logical omniscience is certainly not always an issue, in many applications it is. For example, in the context of distributed computing, we are interested in polynomial-time algorithms, although in some cases the knowledge needed to perform optimally may require calculations that cannot be performed in polynomial time (unless **P=NP**) [Moses and Tuttle 1988]; in the context of security, we may want to reason about computationally bounded adversaries who cannot factor a large composite number, and thus cannot be logically omniscient; in game theory, we may be interested in the impact of computational resources on solution*

> *concepts (for example, what will agents do if computing a Nash equilibrium is difficult). (Halpern & Pucella, 2007)*[23]

El enfoque sintáctico que describe Halpern es similar al de la lógica de justificación. Se trata a la omnisciencia lógica a través de una lista que registra para cada mundo posible las fórmulas que el agente considera verdaderas en un mundo determinado. Describe una estructura sintáctica como una tupla $M = \langle W, W', \pi, C \rangle$, donde $\langle W, W', \pi \rangle$ es un modelo de Kripke convencional, con $W$ un conjunto no vacío de mundos posibles; $W'$ representa las situaciones que el agente considera posibles, es decir, $W' \subseteq W$; $\pi$ es una función de valuación de variables proposicionales; y $C$ es una relación que asocia un conjunto de fórmulas con mundos posibles $w \in W$. La semántica para estas estructuras es la misma que la de la lógica epistémica, excepto que cambia el comportamiento del operador $K$, de modo que:

$(M, w) \models K\varphi$ si y solo si $\varphi \in C(w)$

La relación $C$ es equivalente de la función de evidencia $\mathcal{E}$ de un modelo Kripke-Fitting para lógica de justificación. Halpern no especifica la naturaleza concreta de

[23] Aunque la omnisciencia lógica ciertamente no siempre es un problema, en muchas aplicaciones lo es. Por ejemplo, en el contexto del cómputo distribuido, estamos interesados en algoritmos de tiempo polinomial, aunque en algunos casos el conocimiento necesario para un rendimiento óptimo puede requerir cálculos que no pueden ejecutarse en tiempo polinomial (a menos que $\boldsymbol{P = NP}$) [Moses y Tuttle, 1988]; en el contexto de seguridad, quisiéramos razonar sobre adversarios computacionalmente acotados que no puedan factorizar un número largo no primo, por lo que no pueden ser lógicamente omniscientes; en teoría de juegos, podríamos estar interesados en el impacto de los recursos computacionales disponibles en conceptos de solución (por ejemplo, qué harían los agentes si calcular el equilibrio de Nash es complicado)

esta relación, porque en (Halpern & Pucella, 2007) su intención es recopilar distintas maneras de resolver la omnisciencia lógica. En el siguiente fragmento habla del asunto:

> *[…] if we are using a syntactic structure to represent a given situation, we need to explain where the function C is coming from; with an awareness structure, we must explain where the awareness function is coming from; with an algorithmic knowledge structure, we must explain where the algorithm is coming from; and with an impossible-worlds structure, we must explain what the impossible worlds are.* [24]

La relación *C* que describe es un bosquejo para tratar la evidencia explícitamente, algo que la lógica de justificación resuelve con mucho más detalle. El tratamiento algorítmico que realiza también es similar al análisis de Artemov y Kuznets utilizando complejidad en demostraciones.

[24] […] si estamos usando una estructura sintáctica en una situación particular, necesitamos explicar a qué se refiere la función *C*; con una estructura de percepción, debemos explicar a qué se refiere la función de percepción; con una estructura de conocimiento algorítmico, necesitamos explicar el origen del algoritmo; y con una estructura de mundos imposibles, necesitamos explicar qué son los mundos imposibles.

# 5. Implementación del teorema de elucidación

## 5.1. Introducción

La lógica de justificaciones se deriva de la lógica de demostraciones **LP**, que corresponde a una versión explícita de **S4**. Una fórmula válida en el lenguaje de **LP** es también válida en **S4** a través de lo que se conoce como *proyección de olvido* (*cf. forgetful projection*), que, a partir fórmulas en **LP** con afirmaciones explícitas, se puede encontrar una expresión modal implícita en **S4** equivalente. El paso inverso también es posible, pudiendo encontrar una expresión explícita de fórmulas en **S4** y un polinomio de justificación en **LP**, a este paso se le conoce como *teorema de elucidación* (*cf. Realization Theorem*), con la intuición de que, a partir de una afirmación de conocimiento en **S4** de carácter implícito de la forma $\Box P$ se llega a una fórmula explícita en **LP** con la forma $t{:}\, P$, que se lee «$t$ es una razón para $P$».

*Elucidación* en el sentido de explicación o aclaración, hacer explícita la información que antes no lo era. Un sentido similar puede ser la idea de explicación, por ejemplo la explicación científica, *grosso modo*, utilizar términos familiares para describir un fenómeno no familiar.

El propósito de esta implementación es dar una formalización de un sistema de inferencia al estilo de deducción natural usando lógica de justificación. Debido a la similitud y relación con la lógica modal, también fue necesario formalizar un cálculo de inferencia de lógica modal. La formalización en ambos casos se realizó con el asistente de demostraciones Coq. Un asistente de demostraciones como Coq incluye un sistema de desarrollo de demostraciones que permite interactuar

con métodos de prueba, algoritmos de decisión y tácticas en una demostración implementada en su lenguaje interno *vernacula*, y a través de su lenguaje de especificación formal de alto nivel *Gallina*. COQ también cuenta con un verificador de demostraciones que recibe como entrada la demostración del usuario y responde si es correcta o no, paso a paso de manera interactiva. El verificador de demostraciones de COQ está basado en el isomorfismo Curry-Howard entre cálculo-$\lambda$ tipado y lógica. Esto significa que el verificador recibe la demostración como un término $\lambda$ que corresponde con la prueba. De acuerdo con el isomorfismo Curry-Howard, una demostración $M$ es una demostración de una fórmula $A$ si y solo si el término $\lambda$ correspondiente tiene el tipo $A$. Por lo que COQ es en realidad un verificador de tipos.

La implementación se realizó con la versión 8.4pl4 de COQ. Parte del código fue inspirado por el trabajo de Paulien de Wind sobre deducción natural para lógica modal (de Wind, 2001). En ese trabajo, sin embargo, se utiliza la versión 6.3 de COQ de hace cerca de 14 años, que es muy diferente a las versiones recientes. Se reescribieron las estructuras de datos y demostraciones, pero sirvió como inspiración inicial de esta implementación.

**Cálculo de Construcciones Inductivas**

El lenguaje formal de COQ es una implementación de un cálculo de construcciones con definiciones inductivas. Todos los objetivos en una demostración tienen un tipo. Hay tipos para funciones (o programas), tipos atómicos (tipos de dato), tipos para demostraciones y tipos para identificar al propio tipo. Cualquier objeto que

se manipule utilizando esta formalización debe pertenecer a un tipo. Por ejemplo, la afirmación «para todo *x*, *P*» no está permitida en teoría de tipos, debe plantearse como «para todo *x* que pertenece a tipo *T*, *P*». La expresión «que pertenece al tipo *T*» se escribe en Coq como «*x*: *T*». También se lee como «*x* tiene el tipo *T*».

Los tipos inductivos son ampliamente utilizados para expresar la especificación formal de una teoría matemática en Coq. En la formalización del sistema de inferencia de deducción natural de lógica modal y de justificación se utilizaron tipos inductivos para las definiciones recursivas del lenguaje y de las reglas de inferencia.

***Tipos inductivos***

Un tipo inductivo representa al conjunto más pequeño que es cerrado bajo la aplicación de sus constructores. Un ejemplo habitual en Coq es el de los números naturales. Los números naturales se representan en Coq con el tipo *nat* definido como el conjunto que contiene el constructor «*O*» para cero, y es cerrado bajo el constructor *S* que expresa el sucesor de un miembro del conjunto. En Coq la definición es como sigue:

```
Inductive nat: Set :=
| O : nat
| S : nat -> nat.
```

Esta definición expresa las dos maneras en que un elemento pertenezca al tipo *nat*, o bien es un elemento *O*, o es la aplicación del constructor *S* a un elemento de *nat*.

La documentación oficial de Coq (https://coq.inria.fr/tutorial) ofrece una introducción breve a la sintaxis y comandos básicos del asistente de demostraciones.

La implementación está dividida en dos partes principales. La primera parte especifica un sistema de inferencia de deducción natural para lógica de justificación y lógica modal. La segunda es la implementación semántica de los modelos de Kripke para lógica modal y modelos de Fitting justificados para lógica de justificación.

**Polaridad de aparición de subfórmulas en una fórmula**

Antes de enunciar el teorema de elucidación, se presenta la definición recursiva de la polaridad de las apariciones de una subfórmula en una fórmula (Gabbay & Guenthner, 2010).

i. La aparición de la subfórmula $A$ en $A$ es positiva

ii. Si la polaridad de $A$ es positiva (negativa), en los casos: $(A \wedge B)$, $(B \wedge A)$, $(A \vee B)$, $(B \vee A)$, $(B \rightarrow A)$, $\Box A$, $\Diamond A$, es positiva (negativa). Es decir, no cambia de polaridad.

iii. Si la polaridad de $A$ es positiva (negativa), en los casos $(A \rightarrow B)$ y $\neg A$, es negativa (positiva). Es decir, en estos casos cambia la polaridad de la aparición de $A$.

La aparición de un símbolo $X$ en una fórmula $A$ es de polaridad positiva (negativa), si forma parte de la aparición positiva de alguna subfórmula de $A$, es decir, la polaridad positiva es la que se toma como referencia.

Por ejemplo, si $A$ inicia con polaridad positiva, en la fórmula $A \to B$, la subfórmula $A$ tendrá polaridad negativa. A modo de mnemotecnia, se puede ver como el «signo» que tendría si la fórmula que se está analizando tiene la forma $\varphi \to \psi$, y al escribirla como $\neg\varphi \vee \psi$.

### Teorema de elucidación

El propósito de formular el teorema de elucidación es establecer una relación entre fórmulas que expresan conocimiento utilizando un operador modal implícito y fórmulas que expresen conocimiento de manera explícita con el aparato de la lógica de justificación.

> ***Teorema de elucidación*** *(cf. Realization Theorem) (S. Artemov, 2008).*
> *Si $Z$ es un teorema de* ***S4****, existe una manera de reemplazar las apariciones del operador modal $\Box$ con un polinomio de justificación para producir un teorema demostrable de manera inyectiva en LP. Las apariciones negativas de $\Box$ en $Z$ serán siempre reemplazadas por una variable de justificación distinta, y las apariciones positivas por un polinomio de justificación que involucre esas variables.*

### Variables para términos de justificaciones

Debido a que Coq es un asistente interactivo de demostraciones, la elección de asignar una nueva variable para términos de justificación depende del usuario. En la implementación, esto se expresa con variables indeterminadas definidas en el lenguaje de especificación de Coq.

```
Variable (x y: just).
```

Las variables *x* y *y* del tipo *just* son términos de justificación que se definirán formalmente más adelante. Estas variables se asignarán a términos arbitrarios con los que se está calculando el polinomio de justificación durante la demostración.

**Demostraciones inyectivas y conjunto de especificación constante**

Una demostración inyectiva en una lógica de justificación **L** se refiere a que, por cada miembro del conjunto de especificación contante $\mathcal{CS}$, corresponde una única constante de justificación. En la implementación esto se expresa con constantes particulares asignadas para cada axioma que forma parte de $\mathcal{CS}$:

```
Inductive CS: formula -> Prop :=

| cs1: forall (P Q:formula),
        CS ((c 1):(AxThen1 P Q))

| cs2: forall (P Q R:formula),
        CS ((c 2):(AxThen2 P Q R))

| cs3: forall (P Q:formula),
        CS ((c 3):(AxAnd1 P Q))
…
```

Que es la definición inductiva de $\mathcal{CS}$, y cada $cs_i$ es un elemento particular del conjunto. En este caso *AxThen*1 corresponde con el axioma para la implicación:

```
Definition AxThen1 (P Q:formula) :=
(P --> (Q --> P)).
```

## 5.2. Trabajos relacionados

Melvin Fitting en (Fitting, 2013) desarrolló un mecanismo automático en Prolog para calcular el polinomio de justificación a partir de una fórmula modal en **S4**. La

diferencia entre la implementación de Fitting y la que se presenta en este trabajo, es que el sistema de inferencia en COQ funciona con deducción natural a diferencia de Fitting, que trabaja con tablas semánticas (*cf. semantic tableaus*) con un algoritmo de resolución similar al utilizado por Prolog para lógica de primer orden. Otra diferencia con el trabajo de Fitting, es que la implementación en COQ no obtiene el polinomio de justificación de manera automática. La obtención del polinomio de justificación se valida en cada paso de inferencia de deducción natural con las reglas implementadas.

Al ser deducción natural, las demostraciones para lógica modal corresponden con las presentadas en libros de texto de referencia como (Huth & Ryan, 2004) o (Chellas, 1980), de las que se presentan ejemplos concretos.

## 5.3. S4LP en COQ

Se implementó la lógica de justificación **S4LP** (S. N. Artemov & Nogina, 2005), por contar con un conjunto de axiomas más amplio, que satisface tanto a fórmulas de lógica modal **S4** como fórmulas justificadas en **LP**. Se presentarán los módulos desarrollados, aunque no completos para mejorar la legibilidad del texto. El código fuente de la implementación puede ser descargado de:

https://sites.google.com/site/s4lpcoq

### Sintaxis

Para cada variable proposicional se diseñó un tipo inductivo que toma como argumento un número natural para emular la notación de subíndices: $p_0, p_1, \ldots, p_n$.

```
Inductive var: Set :=
p : nat -> var.
```

Los términos de justificación también se especifican con un tipo inductivo que distingue entre constantes de justificación ($c_0, c_1, \ldots, c_n$), justificaciones arbitrarias para representar variables ($e_0, e_1, \ldots\ e_n$)[25], justificaciones derivadas de los axiomas de suma ($s + t$), de aplicación ($s \cdot t$), y de verificación ($!\,t$).

```
Inductive just: Set :=
| e: nat -> just
| c: nat -> just
| jsum: just->just->just
| japp: just->just->just
| jchk: just->just.
```

Las fórmulas están expresadas con el siguiente tipo inductivo:

```
Inductive formula: Set :=
| Atom : var -> formula
| Ver : formula
| Not : formula -> formula
| If : formula -> formula -> formula
| Box : formula -> formula
| Just : just -> formula -> formula.
```

Donde, *Atom* es un átomo proposicional, *Ver* es la constante de verdad $\top$, *Not* es la negación proposicional $\neg$, *If* es la implicación $\rightarrow$, *Box* el operador modal $\Box$, y *Just* el constructor para una formula justificada $t{:}\,F$.

Las demás operaciones se definen a partir de las anteriores. Esto se hace en Coq de la siguiente manera:

```
Definition Or (f1:formula) (f2:formula): formula :=
(If (If f1 f2) f2).

Definition And (f1:formula) (f2:formula): formula :=
```

[25] *e* de «evidencia»

```
(Not (If f1 (Not f2))).

Definition Fal: formula :=
(Not Ver).

Definition Diam (f:formula): formula :=
(Not (Box (Not f))).

Definition Iif (f1 f2: formula): formula :=
(And (If f1 f2) (If f2 f1)).
```

Donde *Or* es $\vee$, *And* es $\wedge$, *Fal* es $\bot$, *Diam* es $\Diamond$, y *Iif* es $\leftrightarrow$. Esto fundamentado en que $\{\neg, \rightarrow, \Box\}$ es un conjunto funcional completo y las demás operaciones se pueden definir a partir de estos.

Para mejorar la lectura de las fórmulas durante el modo interactivo de demostración, se abrevian estos operadores con la siguiente notación

```
Notation "p /\' q" :=
(And p q) (at level 80, right associativity).

Notation "p \/' q" :=
(Or p q) (at level 85, right associativity).

Notation "¬ p" :=
(Not p) (at level 85, right associativity).

Notation "p --> q" :=
(If p q) (at level 90, right associativity).

Notation "p <--> q" :=
(Iif p q) (at level 60, right associativity).

Notation "[] F" :=
(Box F) (at level 60, right associativity).

Notation "<> F" :=
(Diam F) (at level 60, right associativity).

Notation "s : F":=
(Just s F) (at level 60, right associativity).

Notation "s @ t" :=
(japp s t) (at level 60, right associativity).

Notation "s # t" :=
(jsum s t) (at level 60, right associativity).

Notation "! s " :=
(jchk s) (at level 60, right associativity).
```

Por ejemplo, la fórmula $p \rightarrow q$, se especifica en este lenguaje como:

```
Atom (p 1) --> Atom (p 2)
```

donde $p = p_1$, y $q = p_2$.

La fórmula $\Box\Box(p \rightarrow q)$, sería:

```
[] [] (Atom (p 1) --> Atom (p 2))
```

La fórmula justificada $v{:}\,(p \rightarrow q)$, sería:

```
(e 20):(Atom (p 1) --> Atom (p 2))
```

con la variable de justificación $v$ representada como $e_{20}$.

**Semántica**

Para la tupla que conforma un modelo de Kripke se utilizaron tipos *Record* de COQ, que permiten agrupar diferentes tipos para utilizarlos posteriormente.

```
Record frame:Type := {
  W : Set;
  R : W -> W -> Prop}.

Record kripke: Type := {
  F : frame;
  L : (W F) -> var -> Prop}.

Record fitting: Type := {
  K: kripke;
  E: formula -> just -> (W (F K)) -> Prop}.
```

Aquí, *frame* es el marco que consiste en un conjunto de mundos $\mathcal{W}$ y una relación binaria $\mathcal{R} \subseteq \mathcal{W}\times\mathcal{W}$. El modelo de Kripke está expresado en el tipo *kripke*, que está formado por un marco $\mathcal{F}$ y una función de etiquetación $\mathcal{L}$, que expresa $p \in \mathcal{L}(w)$, donde, $w \in \mathcal{W}$, y $p$ una variable proposicional. El modelo de Fitting es un modelo

de Kripke $\mathcal{K}$ y una función $\mathcal{E}$ que relaciona fórmulas del lenguaje $F$ y justificaciones $t$, con mundos $w$ de $\mathcal{W}$, tal que $w \in \mathcal{E}(F, t)$, si $t$ es una justificación para la fórmula $F$ en el mundo $w$.

Con esto, la satisfacción de una fórmula se define con la función recursiva siguiente:

```
Fixpoint satS4LP (M:fitting) (x:(W (F (K M)))) (phi:formula): Prop :=
match phi with
| Ver =>
        True
| (Atom v) =>
        (L (K M) x v)
| ¬ phi =>
        ~(satS4LP M x phi)
| phi1 --> phi2 =>
        (satS4LP M x phi1) -> (satS4LP M x phi2)
| [] phi =>
        forall y:(W (F (K M))), (R (F (K M)) x y) -> (satS4LP M y phi)
| s : phi =>
        ((E M phi s) x) /\ (forall y:(W (F (K M))), (R (F (K M)) x y) ->
(satS4LP M y phi))
end.
```

Donde, para una llamada a *satS4LP*$(M, x, phi)$, con $M$ un modelo de Fitting, $x \in \mathcal{W}$ y una fórmula *phi* del lenguaje, se tiene:

i. *Ver* ($\top$), corresponde con la constante de verdad de Coq, *True*.

ii. El tipo *Atom* aplicado a un argumento $v$, corresponde con la evaluación de la función $\mathcal{L}$ para ese mismo argumento.

iii. Una fórmula negada (¬) corresponde con la negación interna de Coq (~) para la evaluación de *satS4LP*.

iv. La implicación (-->) corresponde con la implicación interna de Coq (->).

v. El operador modal □ corresponde con la definición usual, para la que la fórmula debe cumplirse para todo mundo accesible.

vi. Una fórmula justificada *t:phi*, agrega a la definición de □ la evidencia *t* con la función $\mathcal{E}$.

Más adelante se ofrecerán ejemplos concretos donde se utiliza la teoría de modelos para fórmulas justificadas. Antes se revisará el sistema de inferencia de deducción natural, que no requiere de evaluación semántica por ser un mecanismo puramente simbólico de razonamiento.

**Axiomatización**

Para trabajar con un conjunto de especificación constante y asignarle un término de justificación a cada axioma, es necesario seleccionar un conjunto de axiomas que permitan deducir las fórmulas con las que se trabajarán. Para una lógica de justificación particular (en este caso **S4LP**) se debe especificar este conjunto $\mathcal{CS}$, de constantes de justificación asociadas. Esto en COQ se implementó con el siguiente tipo inductivo.

```
Inductive CS: formula -> Prop :=

| cs1: forall (P Q:formula),
        CS ((c 1):(AxThen1 P Q))

| cs2: forall (P Q R:formula),
        CS ((c 2):(AxThen2 P Q R))

| cs3: forall (P Q:formula),
        CS ((c 3):(AxAnd1 P Q))

| cs4: forall (P Q:formula),
        CS ((c 4):(AxAnd2 P Q))

| cs5: forall (P Q:formula),
        CS ((c 5):(AxAnd3 P Q))
```

```
| cs6: forall (P Q:formula),
       CS ((c 6):(AxOr1 P Q))

| cs7: forall (P Q:formula),
       CS ((c 7):(AxOr2 P Q))

| cs8: forall (P Q R:formula),
       CS ((c 8):(AxOr3 P Q R))

| cs9: forall (P:formula),
       CS ((c 9):(AxFalse P))

| cs10: forall (s t:just) (P Q: formula),
       CS ((c 10):(AxApp s t P Q))

| cs11: forall (s t:just) (P: formula),
       CS ((c 11):(AxSum1 s t P))

| cs12: forall (s t:just) (P: formula),
       CS ((c 12):(AxSum2 s t P))

| cs13: forall (t:just) (P:formula),
       CS ((c 13):(AxChk t P))

| cs14: forall (t:just) (P:formula),
       CS ((c 14):(AxFact t P))

| cs15: forall (P Q: formula),
       CS ((c 15):(AxK P Q))

| cs16: forall (P: formula),
       CS ((c 16):(Ax4 P))

| cs17: forall (P: formula),
       CS ((c 17):(AxT P))

| cs18: forall (t:just) (P:formula),
       CS ((c 18):(AxS4LP t P))

| cs19: forall (t:just) (P:formula),
       CS ((c 19):(AxS4LPN t P)).
```

A cada axioma se le asignó una constante de justificación particular. En este caso, el conjunto $\mathcal{CS}$ corresponde con el especificado en (S. N. Artemov & Nogina, 2005), pero aquí es necesario utilizar subíndices explícitos. Los axiomas corresponden a los listados en § A.3.

**Reglas de inferencia**

El predicado *Assert* es una formalización de la noción de hipótesis:

```
Parameter Assert: formula->Prop.
```

Para implementar las reglas de inferencia se diseñó un predicado de demostrabilidad, *Provable*. La estructura es también inductiva y es similar a la especificación de las reglas de *introducción* y *eliminación* utilizadas en deducción natural. Por ejemplo, la introducción y la eliminación de implicación, respectivamente, se implementaron de esta manera:

```
Inductive Provable: formula -> Prop :=
...
| impI: forall (phi1 phi2: formula),
      (Assert phi1 -> Provable phi2) ->
      Provable (phi1 --> phi2)

| impE: forall (phi1 phi2: formula),
      Provable (phi1 --> phi2) ->
      Provable phi1 ->
      Provable phi2
...
```

Al conjunto de reglas de inferencia en COQ presentado en (de Wind, 2001) se agregaron las reglas correspondientes a **S4LP**. Por ejemplo, para el axioma de aplicación, se agregó:

```
| justApp: forall (s t:just) (phi1 phi2: formula),
      Provable (s:(phi1 --> phi2)) ->
      Provable (t:phi1) ->
      Provable ((s @ t):phi2)
```

Otra característica importante de este sistema de reglas, es que la *necesitación* se implementó en la versión modal solo para axiomas proposicionales y los axiomas aceptados por la lógica modal particular que caracteriza a la relación de accesibi-

lidad. Para el caso de lógica de justificación, la necesitación corresponde con fórmulas que pertenecen al conjunto de especificación constante $\mathcal{CS}$. Es común tomar este enfoque para el caso de la necesitación en lógica modal, aunque el debate sobre su relación con el teorema de deducción sigue abierto. Para más detalles sobre este tema, se puede consultar (Hakli & Negri, 2012).

La necesitación modal se implementó de la siguiente manera:

```
| modNec: forall (phi:formula),
      AxS4 phi ->
      Provable ([] phi)
```

donde el predicado *AxS4* se define con los axiomas para **S4**, es decir, axiomas proposicionales como *And1;*

```
Definition AxAnd1 (P Q:formula) :=
((P /\' Q) --> P).
```

así como los que caracterizan a una lógica modal S4:

```
Definition AxK (phi psi: formula) :=
[] (phi --> psi) --> [] phi
--> [] psi.

Definition Ax4 (phi: formula) :=
[] phi
--> [] [] phi.

Definition AxT (phi: formula) :=
[] phi
--> phi.
```

Todos estos en la estructura inductiva del predicado AxS4:

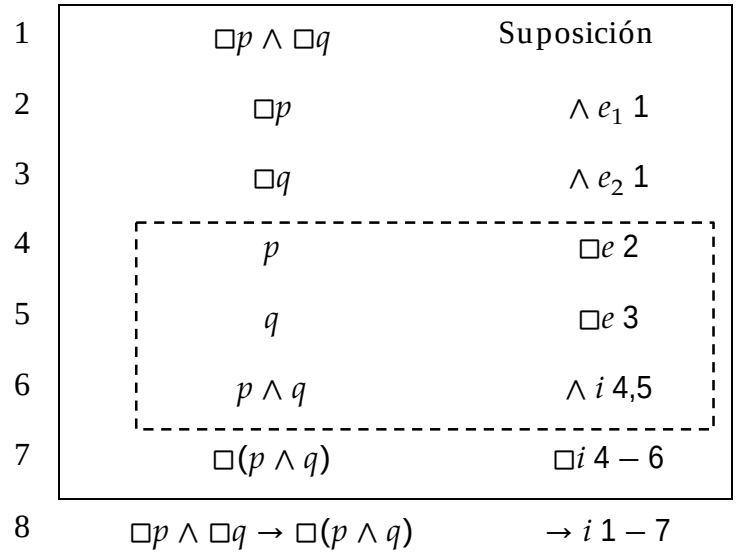
```
Inductive AxS4: formula -> Prop :=
...
| A3S4: forall (P Q:formula),
        AxS4 (AxAnd1 P Q)

...
| A10S4: forall (P Q: formula),
        AxS4 (AxK P Q)

| A11S4: forall (P: formula),
        AxS4 (Ax4 P)
```

```
| A12S4: forall (P: formula),
        AxS4 (AxT P).
```

De esta manera es posible reconocer axiomas durante la demostración.

## 5.4. Ejemplos

En (Huth & Ryan, 2004), en el ejemplo 5.21-1, se presenta la siguiente demostración en la lógica **K** de la fórmula $\vdash_K \Box p \wedge \Box q \rightarrow \Box(p \wedge q)$.




En el sistema de deducción natural implementado en COQ esta demostración se ve de la siguiente manera:

```
Example HuthRyan521_1: forall (P Q:formula),
Provable (([] P /\' [] Q) --> [] (P /\' Q)).
Proof.
intros.
apply impI; intros.

eapply modKE.
eapply modKE.
apply modNec.
apply A5S4.

apply ass in H.
eapply andE1 in H.
trivial.
```

```
apply ass in H.
eapply andE2 in H.
trivial.
Qed.
```

Donde, *modKE* es análoga a *modus ponens* pero para el operador modal □. Las dos aplicaciones de la regla *modKE* resultan en la siguiente fórmula

```
P : formula
Q : formula
H : Assert ([]P /\' []Q)
============================================================================
(1/3)
Provable ([](?416 --> ?413 --> P /\' Q))
```

Es decir, al llamar a *eapply modKE*, COQ le da una expresión indeterminada (expresada con ?416 y ?413, pero son valores numéricos arbitrarios que expresan variables nuevas) al antecedente de una implicación que se refiere al axioma **K**, en este caso.

A continuación, se aplica el predicado A5S4, que se refiere a que COQ va a empatar la fórmula objetivo con el axioma 5 de **S4**.

```
| A5S4: forall (P Q:formula),
        AxS4 (AxAnd3 P Q)
```

que es una implementación del axioma proposicional And3:

```
Definition AxAnd3 (P Q:formula) :=
(P --> (Q --> (P /\' Q))).
```

El resto de la demostración es rutinario, COQ pide la demostración de $\Box P$ y $\Box Q$, que se obtienen de la suposición inicial $\Box P \wedge \Box Q$.

Se implementaron también las demostraciones que se utilizan de ejemplo para mostrar deducción natural en **S5** en ese mismo capítulo:

```
Example HuthRyan521_2: forall (P: formula),
Provable (P --> ([] <> P)).

Example HuthRyan521_3: forall (P:formula),
Provable (([] <> [] P) --> ([] P)).
```

Así como un par de ejercicios del libro (Chellas, 1980), sección 8.20.

```
Example Chellas820_1: Provable ((<> Ver) <--> (¬ [] Fal)).
Example Chellas820_2: Provable (([] Ver) <--> (¬ <> Fal)).
```

## 5.5. Deducción natural con lógica de justificación

### Granero rojo

Como ejemplo de utilización del sistema de inferencia por deducción natural con lógica de justificación, se usará el ejemplo de los graneros rojos visto anteriormente. Recapitulando, se tiene la siguiente formalización para una interpretación doxástica:

4. $\Box Ba$, «creo que el objeto frente a mí es un granero»;

5. $\Box(Ba \wedge Re)$, «creo que el objeto frente a mí es un granero rojo».

Donde, *Ba* se refiere a «*barn*» y *Re* a «*red*». Se utilizan estos nombres de variables para no entrar en conflicto con otros elementos del programa en Coq, en particular, *R* se utiliza más adelante para la relación de accesibilidad.

Artemov en (S. Artemov, 2008) menciona que si se formaliza el mismo ejemplo con una lógica de conocimiento, como S4, partiendo de la suposición externa de que $\neg\Box Ba$, porque la creencia del agente no puede constituir conocimiento (por haber graneros falsos) y es posible derivar $\Box Ba$, la lógica modal epistémica estaría permitiendo concluir un absurdo.

Esto en el sistema de deducción implementado se ve así:

```
Example GranerosRojosK: forall (Ba Re:formula),
    Provable((¬[]Ba) --> ([](Ba /\' Re)) --> ([]Ba)).
intros.
apply impI; intro.
apply impI; intro.

eapply modKE.
apply modNec.
apply A3S4.
apply ass.
exact H0.
Qed.
```

Es decir, a partir de la hipótesis $\neg\Box Ba$ y $\Box(Ba \wedge Re)$, permite concluir $\Box Ba$. Artemov hace esta deducción usando el hecho de que $(Ba \wedge Re \rightarrow Ba)$ es una tautología, esto mismo se observa en la demostración con COQ, empleando el predicado A3S4, que corresponde con la misma tautología.

Artemov en este artículo utiliza lógica de justificación y sostiene que no se llega a una contradicción porque usar justificaciones explícitas permite derivar sin absurdos $\neg u{:}\, Ba$ y $(a \cdot v){:}\, Ba$ por ser justificaciones distintas. En COQ esta derivación corresponde con:

```
Example GranerosRojos: forall (u v:just) (Ba Re:formula), exists (a:just),
    Provable((¬u:Ba) --> (v:(Ba /\' Re)) --> ((a @ v):Ba)).
Proof.
intros.
eexists.
apply impI; intro.
apply impI; intro.
eapply justApp.
apply justNec.
apply cs3.
apply ass.
exact H0.
Qed.
```

Se emplea el cuantificador existencial para la justificación (*a:just*) para establecer en el aparato formal que se trata de una justificación indeterminada. La demostración es análoga a la presentada para **S4**. Se utiliza ahora la necesitación justificada

*justNec*, que utiliza el conjunto de especificación constante $\mathcal{CS}$, en la aplicación de *cs3*, que corresponde también con la tautología $Ba \wedge Re \rightarrow Ba$, pero con una justificación constante asignada, $c_3$:

```
| cs3: forall (P Q:formula),
       CS ((c 3):(AxAnd1 P Q))
```

**Construcción de polinomios de justificación**

La idea principal detrás del teorema de elucidación es que se pueden construir polinomios de justificación para fórmulas modales válidas. El ejemplo 3.1 de Artemov en (S. Artemov, 2008) hace precisamente eso, con la siguiente fórmula modal que relaciona la *conjunción* entre dos fórmulas.

$$\Box A \wedge \Box B \rightarrow \Box(A \wedge B)$$

La construcción que ofrece Artemov con un razonamiento línea a línea en ese artículo es la siguiente:

1. $A \rightarrow (B \rightarrow (A \wedge B))$, un axioma proposicional;
2. $c_5 : [A \rightarrow (B \rightarrow (A \wedge B))]$, de 1, por internalización;
3. $x : A \rightarrow (c_5 \cdot x) : (B \rightarrow (A \wedge B))$, de 2 por el axioma de aplicación y modus ponens;
4. $x : A \rightarrow (y : B \rightarrow ((c_5 \cdot x) \cdot y) : (A \wedge B))$, de 3, por el axioma de aplicación y razonamiento proposicional;
5. $x : A \wedge y : B \rightarrow ((c_5 \cdot x) \cdot y) : (A \wedge B)$, de 4, por razonamiento proposicional.

Como se puede apreciar, en esta construcción de la justificación de la conjunción se respeta el teorema de elucidación y en las apariciones negativas de las subfórmulas $A$ y $B$ se asignan variables de justificación y la justificación de la conjunción es un polinomio que involucra esas variables, considerando a la constante $c$ para el axioma proposicional $A \to (B \to (A \wedge B))$.

En COQ esta demostración se realiza de la siguiente manera:

```
Example andJust: forall (A B: formula), exists (a b c: just),
Provable(a:A /\' b:B --> c:(A /\' B)).
Proof.
intros.
exists x.
exists y.
eexists.

apply impI; intro.

eapply justApp.
eapply justApp.

eapply justNec.
apply cs5.

apply ass in H.
apply andE1 in H.
exact H.

apply ass in H.
apply andE2 in H.
exact H.

Qed.
```

Como se puede observar, la demostración en COQ es análoga a la ofrecida por Artemov. La constante de justificación en este caso es $c_5$, que corresponde en $\mathcal{CS}$ con la misma tautología proposicional. Para las justificaciones indeterminadas $a$, $b$ y $c$, se asignan variables específicas para las dos primeras ($x$ y $y$) y se calcula el resultado para la última.

Del mismo modo, se construye la justificación para la *disyunción* de dos fórmulas. La fórmula modal que se va a elucidar es la siguiente:

$$\Box A \vee \Box B \rightarrow \Box(A \vee B)$$

Artemov da la siguiente manera de construir la justificación de la disyunción:

1. $A \rightarrow (A \vee B)$, axioma proposicional;
2. $c_6 : [A \rightarrow (A \vee B)]$, de 1, por internalización;
3. $x : A \rightarrow (c_6 \cdot x) : (A \vee B)$, de 2, por el axioma de aplicación y modus ponens;
4. $B \rightarrow (A \vee B)$, axioma proposicional;
5. $c_7 : [B \rightarrow (A \vee B)]$, de 4, por internalización;
6. $y : B \rightarrow (c_7 \cdot y) : (A \vee B)$, de 5, por el axioma de aplicación y modus ponens;
7. $(c_6 \cdot x) : (A \vee B) \rightarrow (c_6 \cdot x + c_7 \cdot y) : (A \vee B)$, por el axioma de suma;
8. $(c_7 \cdot y) : (A \vee B) \rightarrow (c_6 \cdot x + c_7 \cdot y) : (A \vee B)$, por el axioma de suma;
9. $(x : A \vee y : B) \rightarrow (c_6 \cdot x + c_7 \cdot y) : (A \vee B)$, de 3, 6, 7, 8 por razonamiento proposicional.

En COQ esta construcción quedaría:

```
Example orJust: forall (A B: formula), exists (a b c: just),
Provable(a:A \/' b:B --> c:(A \/' B)).
Proof.
intros.
exists x.
exists y.
```

```
eexists.

apply impI; intro.

assert (Assert (x:A) -> Provable(((c 6 @ x) # c 7 @ y) : (A \/' B))).
intro.
apply justS1.
eapply justApp.
apply justNec.
apply cs6.
apply ass; trivial.

assert (Assert (y:B) -> Provable(((c 6 @ x) # c 7 @ y) : (A \/' B))).
intro.
apply justS2.
eapply justApp.
apply justNec.
apply cs7.
apply ass; trivial.

eapply orE.

apply ass; exact H.

exact H0.
exact H1.

Qed.
```

Donde la constante de justificación $c_6$ corresponde con la justificación en $\mathcal{CS}$ de la tautología proposicional $P \rightarrow (P \vee Q)$, y $c_7$ con la justificación de $Q \rightarrow (P \vee Q)$. La demostración corresponde con la ofrecida por Artemov. De este modo podemos ver que la formalización en COQ funciona correctamente.

## 5.6. Demostración en COQ de propiedades interesantes

### Introspección positiva derivable en S4LP

Artemov y Nogina en (S. N. Artemov & Nogina, 2005) introducen la lógica de justificación a un contexto epistémico relacionando la lógica epistémica convencional **S4** con la lógica de justificación que le corresponde, **LP**. A esta lógica combinada se le conoce como **S4LP**, que simplemente es una lógica que conecta a ambas a

través del principio de unión entre el conocimiento implícito expresado por **S4** y el explícito por **LP**.

$$t{:}F \rightarrow \Box F$$

para algún término de justificación $t$ y una fórmula $F$.

En este artículo se ofrece una demostración de que el principio de introspección positiva es derivable en **S4LP** con un conjunto de especificación constante vacío. La demostración es bastante sencilla:

En $S4LP_{\emptyset}$:

1. $t{:}F \rightarrow !t{:}(t{:}F)$, instancia del axioma de verificación;
2. $!t{:}(t{:}F) \rightarrow \Box t{:}F$, por el axioma de conexión entre **S4** y **LP**;
3. $t{:}F \rightarrow \Box t{:}F$, por razonamiento proposicional.

■

En la implementación en COQ, tenemos:

```
Lemma positiveInt: forall (t:just) (P: formula),
Provable (t:P) -> Provable([]t:P).
Proof.
intros.

eapply justS4LP.

apply justChk.
trivial.
Qed.
```

Que corresponde con la demostración de Artemov y Nogina.

**Decidibilidad de la evidencia**

Aceptando un axioma adicional, la introspección negativa explícita:

$$\neg t{:}F \rightarrow \Box \neg t{:}F$$

Se define la lógica **S4LPN**, con la que se puede demostrar el principio de decidibilidad de la evidencia (S. N. Artemov & Nogina, 2005).

$$\Box t{:}F \vee \Box \neg t{:}F$$

Demostración. Razonando en S4LPN:

1. $t{:}F \rightarrow \Box t{:}F$, una instancia de la introspección positiva;
2. $\neg t{:}F \rightarrow \Box \neg t{:}F$, una instancia de la introspección negativa;
3. $(t{:}F \vee \neg t{:}F) \rightarrow (\Box t{:}F \vee \Box \neg t{:}F)$, de 1 y 2, por lógica proposicional;
4. $\Box t{:}F \vee \Box \neg t{:}F$ , por lógica proposicional, considerando como axioma $\varphi \vee \neg \varphi$.

■

Considerando que el aparato deductivo es *decidible* en Coq, la misma demostración se presenta a continuación. Es importante notar que hasta el momento no se había considerado como axioma $\varphi \vee \neg \varphi$, por lo que solo se utiliza para este caso, aceptando como premisa *decProv*.

```
Section ProvDec.

Hypotheses decProv: forall (phi:formula),
Provable (phi \/' ¬phi).

Lemma decEvidence: forall (t:just) (P:formula),
Provable ([]t:P \/' []¬t:P).
Proof.
intros.

assert (Assert(t:P) -> Provable([]t:P \/' []¬t:P)).
intro.
apply orI1.
apply positiveInt.
apply ass; trivial.

assert(Assert(¬t:P) -> Provable([]t:P \/'[]¬t:P)).
intro.
apply orI2.
apply justS4LPN.
apply ass; trivial.

eapply orE.
apply decProv.
apply H.
trivial.

Qed.

End ProvDec.
```

### **Axioma** K **en lógica de justificación**

En el capítulo «Conocimiento y Justificación» de este trabajo, se mencionó que se puede construir un modelo en el que no se cumpla el axioma de distribución. Sin embargo, partiendo de las operaciones entre términos de justificación, si se postula la siguiente propiedad para la operación aplicación (·)

*Para todo término de justificación* $t$ *y toda formula* $F$*, si* $(t \cdot t) : F$*, entonces* $t : F$

Es decir, un sentido de idempotencia para (·), en el que se anula el efecto de la aplicación de un término de justificación $t$ dos veces a la misma fórmula.

Esta sentido de idempotencia, permitía que se cumpliera la propiedad de cerradura bajo implicación lógica:

$$r{:}P \rightarrow r{:}(P \rightarrow Q) \rightarrow r{:}Q$$

La demostración en COQ de esta característica es la siguiente:

```
Section AppIdempotent.
Hypotheses idempot: forall (t:just) (phi:formula),
Provable((t@t):phi) -> Provable (t:phi).

Lemma idemMod: forall (P Q: formula), exists (t:just),
Provable (t:P --> t:(P --> Q) --> t:Q).
Proof.
intros.
exists x.
apply impI; intro.
apply impI; intro.

apply idempot.
eapply justApp.

apply ass; exact H0.
apply ass; exact H.
Qed.

End AppIdempotent.
```

## 5.7. Demostración de Internalización para un $\mathcal{CS}$

Un conjunto de especificación constante $\mathcal{CS}$ es *axiomáticamente apropiado,* si cumple con la siguiente propiedad:

Para cada axioma $A$ existe una constante $e_1$, tal que $e_1{:}A$ está en $\mathcal{CS}$, y si:

$$e_n{:}e_{n-1}{:}\ldots{:}e_1{:}A \in \mathcal{CS}$$

entonces

$$e_{n+1}{:}e_n{:}e_{n-1}{:}\ldots{:}e_1{:}A \in \mathcal{CS}$$

Es decir, cada miembro de $\mathcal{CS}$ también puede tener una justificación constante.

La propiedad de internalización en lógica de justificación es análoga a la necesitación en lógica modal. Es la característica que permite obtener contantes de justificación de fórmulas derivables en el sistema de inferencia. La regla de internalización establece lo siguiente:

*Si* $\vdash F$, *entonces* $\vdash s{:}\, F$, *para algún término de justificación* $s$

La demostración que ofrece Artemov en (S. Artemov, 2008) para una lógica de justificación $J$ con un conjunto de especificación constante $\mathcal{CS}$ axiomáticamente apropiado, es por inducción en la longitud de la derivación.

*Caso base*. Suponiendo $\vdash F$, si $F$ es un axioma, y como $\mathcal{CS}$ es axiomáticamente apropiado, existe una constante $e$, tal que $e{:}\, F \in \mathcal{CS}$, por lo tanto, un axioma de $J_{\mathcal{CS}}$.

*Hipótesis de inducción (H.I.).* Si $F$ pertenece a $\mathcal{CS}$, como $\mathcal{CS}$ es axiomáticamente apropiado, $e{:}\, F \in \mathcal{CS}$.

*Caso inductivo.* Si $F$ se obtuvo de aplicar modus ponens en un paso anterior de $\chi \to F$ y $\chi$, entonces por H.I., $\vdash s{:}\, (\chi \to F)$ y $\vdash t{:}\, \chi$, para dos justificaciones $s$ y $t$. Por el axioma de aplicación se puede construir la justificación para $F$, y resulta en $(s \cdot t){:}\, F$.

■

En COQ, la propiedad de un $CS$ de ser axiomáticamente apropiado se implementó con:

```
Hypotheses CSapropiate:
forall (phi:formula),
Provable phi -> exists t, CS (t:phi).
```

La demostración de la internalización en COQ, también por inducción en la derivabilidad de la fórmula, queda de la siguiente manera:

```
Theorem Internalization (phi:formula):
Provable (phi) -> exists (t:just), (Provable (t:phi)).
Proof.
intros.

induction H.
destruct (CSapropiate Ver).
apply truth.
eexists.
apply justNec.
eexact H.


destruct (CSapropiate phi).
apply ass.
trivial.
eexists.
apply justNec.
eexact H0.


destruct (CSapropiate (phi1 /\' phi2)).
apply andI; trivial; trivial.
eexists.
apply justNec.
eexact H1.


destruct (CSapropiate phi1).
apply andE1 in H; trivial.
eexists.
apply justNec.
eexact H0.


destruct (CSapropiate phi2).
apply andE2 in H; trivial.
eexists.
apply justNec.
eexact H0.


destruct (CSapropiate (phi1 \/' phi2)).
apply orI1; trivial.
eexists.
```

```
apply justNec.
eexact H0.


destruct (CSapropiate (phi1 \/' phi2)).
apply orI2; trivial.
eexists.
apply justNec.
eexact H0.


destruct (CSapropiate psi).
eapply orE.
eexact H.
trivial.
trivial.
eexists.
apply justNec.
eexact H4.


destruct (CSapropiate (phi1 --> phi2)).
apply impI.
trivial.
eexists.
apply justNec.
eexact H1.


destruct (CSapropiate phi2).
eapply impE.
eexact H.
trivial.
eexists.
apply justNec.
eexact H1.


destruct (CSapropiate (¬phi)).
apply notI.
trivial.
eexists.
apply justNec.
eexact H1.


destruct (CSapropiate Fal).
eapply notE.
eexact H.
trivial.
eexists.
apply justNec.
eexact H1.


destruct (CSapropiate phi).
apply botE.
trivial.
eexists.
apply justNec.
eexact H0.
```

```
destruct (CSapropiate phi).
apply notnotE.
trivial.
eexists.
apply justNec.
eexact H0.


destruct (CSapropiate ([]phi2)).
eapply modKE.
exact H.
trivial.
eexists.
apply justNec.
eexact H1.


destruct (CSapropiate ([](phi1 --> phi2))).
apply modKI.
trivial.
eexists.
apply justNec.
exact H1.


destruct (CSapropiate ([]phi)).
apply modNec.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate phi).
apply modT.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ([][]phi)).
apply mod4.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ([](¬[]phi))).
apply mod5.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ((s@t):phi2)).
eapply justApp.
```

```
exact H.
exact H0.
eexists.
apply justNec.
exact H1.


destruct (CSapropiate ((s#t):phi)).
apply justS1.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ((s#t):phi)).
apply justS2.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ((!t):t:phi)).
apply justChk.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate phi).
apply justNec.
trivial.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ([]phi)).
eapply justS4LP.
exact H.
eexists.
apply justNec.
exact H0.


destruct (CSapropiate ([](¬t:phi))).
apply justS4LPN.
trivial.
eexists.
apply justNec.
exact H0.
Qed.
```

Naturalmente, esta demostración es mucho más extensa que la de Artemov porque en su aparato solo se incluye *modus ponens* como regla de inferencia. Debe considerarse que el predicado de derivabilidad *Provable* que se implementó tiene, además de *modus ponens*, otras reglas de inferencia, pero únicamente para mejorar la legibilidad de las demostraciones.

**Ejemplos semánticos**

Hasta el momento no ha intervenido una interpretación semántica de las fórmulas de lógica de justificación. A continuación se presenta la manera de trabajar semánticamente con la implementación en Coq que se desarrolló.

Para expresar un conjunto de mundos $\mathcal{W}'$, se utiliza la siguiente estructura inductiva:

```
Inductive W':Set :=
| w1:W'
| w2:W'
| w3:W'
| w4:W'.
```

Es decir, $\mathcal{W}' = \{w_1, w_2, w_3, w_4\}$.

Para la relación de accesibilidad $\mathcal{R}'$:

```
Definition R' (x y:W') : Prop :=
match x with
| w1 => match y with
       | w1 => True
       | w3 => True
       | _ => False
       end
| w2 => False
| w3 => match y with
       | w2 => True
       | w4 => True
       | _ => False
       end
```

```
| w4 => False
end.
```

Que representa, $\mathcal{R}' = \{(w_1, w_1), (w_1, w_3), (w_3, w_2), (w_3, w_4)\}$.

Los tipos *Record* en Coq definen un predicado especial para construir nuevos objetos de ese tipo, en este caso es el predicado *Build_frame*, para construir un objeto de tipo *frame*, $\mathcal{F}'$ con los conjuntos que se definieron arriba.

```
Definition F':=(Build_frame W' R').
```

Que representa al marco $\mathcal{F}' = \langle \mathcal{W}', \mathcal{R}' \rangle$.

De manera análoga se definen la función de etiquetación $\mathcal{L}'$:

```
Definition L' (x:W') (pv:var) :=
match x with
| w1 => match pv with
       | (p 0) => True
       | _ => False
       end
| w2 => False
| w3 => match pv with
       | (p 0) => True
       | (p 1) => True
       | _ => False
       end
| w4 => match pv with
       | (p 1) => True
       | _ => False
       end
end.
```

Que representa $\mathcal{L}' = \{(w_1, p_0), (w_3, p_0), (w_3, p_1), (w_4, p_1)\}$.

El modelo de Kripke, $\mathcal{K}'$:

```
Definition K' := (Build_kripke F' L').
```

Para el modelo de Kripke $\mathcal{K}' = \langle \mathcal{F}', \mathcal{L}' \rangle$.

Una función de evidencia $\mathcal{E}'$ que relaciona términos de justificación con fórmulas.

```
Definition E' (phi:formula) (j:just) (x:W') :=
match phi with
 | Atom (p 0) => match j with
              | e 0 => match x with
                       | w1 => True
                       | _  => False
                       end
              | _ => False
              end
 | _ => False
end.
```

Que representa $\mathcal{E}'(e_0, p_0) = \{w_1\}$.

Y finalmente, el modelo de Fitting justificado, $\mathcal{M}'$:

```
Definition M' := (Build_fitting K' E').
```

Es decir, $\mathcal{M}' = \langle \mathcal{K}', \mathcal{E}' \rangle$.

El modelo $\mathcal{M}'$ es uno particular, para el que se puede evaluar si satisface fórmulas específicas. Por ejemplo, para evaluar $\mathcal{M}', x_1 \vDash e_0{:}p_0$, es decir, la variable proposicional $p_0$ justificada con el término $e_0$, en el mundo $x_1$.

```
Example ejMx1: (satS4LP M' w1 ((e 0):(Atom (p 0)))).
unfold satS4LP.
compute.
split.
trivial.
intros.
trivial.
Qed.
```

Lo que en efecto se cumple por las condiciones impuestas por el modelo.

# A. Lógica modal

A continuación se hace un breve repaso de propiedades importantes de la lógica modal que se usan en este trabajo.

## A.1. Lenguaje de la lógica modal

A.1.1. **Definición**. Una fórmula $\varphi$ en lógica modal está dada por la siguiente gramática, donde $p_i$ es un átomo proposicional ($p_0, p_1, ...$), y $\top$ es la constante verdadero:

$$\varphi ::= p_i \mid \top \mid \neg\varphi \mid \varphi \to \varphi \mid \Box\varphi$$

Las demás conectivas proposicionales se pueden definir con la implicación y negación, de la siguiente manera:

- $\bot \equiv \neg\top$
- $\varphi \vee \psi \equiv (\varphi \to \psi) \to \psi$
- $\varphi \wedge \psi \equiv \neg(\varphi \to \neg\psi)$
- $\varphi \leftrightarrow \psi \equiv (\varphi \to \psi) \wedge (\psi \to \varphi)$

Para el dual del operador modal $\Box$, se utilizará $\Diamond$, es decir:

$$\Box\varphi \equiv \neg\Diamond\neg\varphi$$

## A.2. Semántica de la lógica modal

La semántica usual para la lógica modal está dada a través de modelos de Kripke de mundos posibles. Un *modelo de Kripke* es una tupla $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{V} \rangle$, tal que:

- $\mathcal{W}$ es un conjunto no vacío de mundos posibles (estados, situaciones, posibilidades);
- $\mathcal{R} \subseteq \mathcal{W} \times \mathcal{W}$ es una relación binaria que indica qué mundos están accesibles a otros;
- $\mathcal{V}: \mathcal{W} \rightarrow \wp(PROP)$, la función de etiquetación, cuyo dominio es $\mathcal{W}$ y codiminio el conjunto potencia de los átomos proposicionales, tal que $PROP = \{p_0, p_1, \ldots, p_n\}$.

Sea $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{V} \rangle$ un *modelo de Kripke*, la verdad de la fórmula $\varphi$ dada por (A.1.1) en el mundo $w \in \mathcal{W}$, que se escribe como $\mathcal{M}, w \models \varphi$, y está dada por:

Para cada $w \in \mathcal{W}$,

6. $\mathcal{M}, w \models p$, ssi $p \in \mathcal{V}(w)$, para una variable proposicional *p*;
7. $\mathcal{M}, w \models \top$ siempre es el caso;
8. $\mathcal{M}, w \models \neg\varphi$, ssi no es el caso que $\mathcal{M}, w \models \varphi$, es decir, $\mathcal{M}, w \not\models \varphi$;
9. $\mathcal{M}, w \models (\varphi \rightarrow \psi)$, ssi $\mathcal{M}, w \models \varphi$ implica $\mathcal{M}, w \models \psi$;
10. $\mathcal{M}, w \models \Box\varphi$, ssi para cada $v \in \mathcal{W}$, tal que si $wRv$, entonces $\mathcal{M}, v \models \varphi$.

## A.3. Axiomas de la lógica modal

La lógica modal proposicional acepta los axiomas tradicionales de la lógica proposicional. Para este trabajo se consideran axiomas de la lógica proposicional (Troelstra & Schwichtenberg, 2000, p. 51), los siguientes[26]:

A.3.1. *Then1*. $P \to (Q \to P)$

A.3.2. *Then2*. $(P \to (Q \to R)) \to ((P \to Q) \to (P \to R))$

A.3.3. *And1*. $(P \wedge Q) \to P$

A.3.4. *And2*. $(P \wedge Q) \to Q$

A.3.5. *And3*. $P \to (Q \to (P \wedge Q))$

A.3.6. *Or1*. $P \to (P \vee Q)$

A.3.7. *Or2*. $Q \to (P \vee Q)$

A.3.8. *Or3*. $(P \to R) \to ((Q \to R) \to (P \vee Q) \to R)$

A.3.9. *False*. $\bot \to P$

[26] Se consideran solo los axiomas proposicionales. Los identificadores (Then1, Then2…) corresponden con los asignados en la implementación en COQ.

La lógica modal normal acepta como mínimo el axioma **K** (en honor a Saul Kripke), también conocido como *axioma de distribución*:

A.3.10. **K**. $\Box(P \rightarrow Q) \rightarrow (\Box P \rightarrow \Box Q)$

La lógica **KT4**, también conocida como **S4**, acepta además:

A.3.11. **T**. $\Box P \rightarrow P$

A.3.12. **4**. $\Box P \rightarrow \Box\Box P$

La lógica **KT45**, o **S5**, además de los anteriores, acepta:

A.3.13. **5**.$\Diamond P \rightarrow \Box\Diamond P$

Estos axiomas también caracterizan a la relación de accesibilidad $\mathcal{R}$ del modelo de Kripke $\mathcal{M}$, como se muestra en la siguiente tabla.

| Axioma | Fórmula | Propiedad en $\mathcal{R}$ |
|---|---|---|
| **K** | $\Box(P \rightarrow Q) \rightarrow (\Box P \rightarrow \Box Q)$ | Irrestricta |
| **T** | $\Box P \rightarrow P$ | Reflexividad |
| **4** | $\Box P \rightarrow \Box\Box P$ | Transitividad |
| **B** | $P \rightarrow \Box\Diamond P$ | Simetría |
| **D** | $\Box P \rightarrow \Diamond P$ | Serialidad |
| **5** | $\Diamond P \rightarrow \Box\Diamond P$ | Euclidiana |

*Tabla 2. Propiedades de la relación de accesibilidad respecto a fórmulas válidas*

Es decir, si se acepta el axioma $\Box P \rightarrow P$, se está condicionando a $\mathcal{R}$ a ser reflexiva. Por ejemplo, que $\mathcal{R}$ cumpla la propiedad de ser *reflexiva*, significa que para todo $w \in \mathcal{W}$, es el caso que $wRw$. Del mismo modo, si se acepta $\Box P \rightarrow \Box\Box P$ como axioma, $\mathcal{R}$ debe ser *transitiva*, es decir, que para todo $u, v, w \in \mathcal{W}$, si es el caso que

*uRv* y *vRw*, entonces *uRw*. De manera análoga para el resto de las fórmulas de la tabla.

Además de estos axiomas, para el caso de la lógica epistémica, se consideran los siguientes axiomas.

| Axioma | Fórmula |
|---|---|
| **.2** | $\neg\Box\neg\Box P \rightarrow \Box\neg\Box\neg P$ |
| **.3** | $\Box(\Box P \rightarrow \Box Q) \vee \Box(\Box Q \rightarrow \Box P)$ |
| **.4** | $P \rightarrow (\neg\Box\neg\Box P \rightarrow \Box P)$ |

*Tabla 3. Axiomas adicionales para lógica epistémica.*

Estos axiomas describen diferentes aparatos de lógica epistémica. Se considera a **T** (que acepta a los axiomas **K** y **T**) como el sistema más débil de lógica epistémica. A continuación se describen los diferentes sistemas epistémicos de acuerdo a los axiomas que aceptan.

| Sistema epistémico | También conocido como: |
|---|---|
| **KT4** | **S4** |
| **KT4 + .2** | **S4.2** |
| **KT4 + .3** | **S4.3** |
| **KT4 + .4** | **S4.4** |
| **KT5** | **S5** |

*Tabla 4. Sistemas epistémicos de acuerdo a los axiomas que consideran válidos.*

La aceptación de cada axioma le otorga diferentes capacidades al sistema epistémico. No hay una única lógica epistémica, pero en general se aceptan estas propiedades como estándar y varían de acuerdo al propósito del investigador. Por ejemplo, para una lógica de creencias se suele sustituir el axioma **T** por **D**, para evitar la condición de verdad para las creencias pero conservando la consistencia entre un conjunto de creencias.

## A.4. Reglas de inferencia

Normalmente se consideran solo dos reglas de inferencia para lógica modal.

Modus Ponens (**MP**)

$$\frac{\varphi \rightarrow \psi,\ \varphi}{\psi}\ \mathrm{MP}$$

Necesitación (**NEC**)

$$\frac{\varphi}{\Box\varphi}\ NEC$$

# 6. Conclusiones y trabajo futuro

La lógica de justificación ofrece un aparato deductivo tan capaz y maduro como el de la lógica modal, pero permite además hablar de razones para las fórmulas. Hemos revisado un panorama amplio del alcance y relevancia del estudio de la lógica de justificación. El propósito fue dar un recorrido por las áreas en la que ya ha tenido impacto una formulación como esta y plantear posibles soluciones a nuevos problemas, ahora con este nuevo enfoque.

La lógica de justificación es una reforma a las ideas planteadas por la lógica modal. Las conexión entre estas dos formalizaciones se hace evidente con las relaciones entre fórmulas de ambos lenguajes, que se expresan con el teorema de elucidación, en un sentido, y con la proyección de olvido, por el otro. Establecer la relación entre fórmulas modales y de justificación es fundamental para que los términos de justificación no se interpreten simplemente como elementos sintácticos sin contenido, sino que cumplan su papel de conectar fórmulas implícitas con su justificación.

Desarrollar una implementación computacional del cálculo inferencial con justificaciones cumplió el propósito de ejemplificar la factibilidad de nuevas aplicaciones tecnológicas que pueden verse beneficiadas con esta formalización. Que las demostraciones en COQ correspondan con las tradicionales no es coincidencia ni que la implementación se haya hecho *ad hoc*; COQ es una potente implementación de una lógica de orden superior que valida computacionalmente los resultados obtenidos. Las estructuras de datos y cálculo de inferencia utilizados en esta implementación pueden servir de punto de partida para otros proyectos.

Una posible aplicación a futuro puede estar en el problema de representación del conocimiento en inteligencia artificial (Russell & Norvig, 2003, p. 453), porque enriquece el aparato modal con términos de justificación y permite hacer un diagnóstico más detallado de los sistemas omniscientes.

# Referencias


Arora, S., & Barak, B. (2009). *Computational Complexity: A Modern Approach* (1st ed.). New York, NY, USA: Cambridge University Press.

Artemov, S. (2008). THE LOGIC OF JUSTIFICATION. *The Review of Symbolic Logic*, *1*(04), 477–513.

Artemov, S., & Fitting, M. (2012). Justification Logic. En E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2012).

Artemov, S., & Kuznets, R. (2006). Logical omniscience via proof complexity. En *In Computer Science Logic 2006, Lecture Notes in Computer Science, Vol 4207* (pp. 135–149). Springer.

Artemov, S. N. (1998). *Explicit Modal Logic* (No. CFIS 98–17). Cornell University.

Artemov, S. N., & Nogina, E. (2005). Introducing Justification into Epistemic Logic. *Journal of Logic and Computation*, *15*(6), 1059–1073.

Chellas, B. F. (1980). *Modal Logic: An Introduction*. Cambridge University Press.

Cook, S., & Reckhow, R. (1974). On the Lengths of Proofs in the Propositional Calculus (Preliminary Version). En *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing* (pp. 135–148). New York, NY, USA: ACM.

D'Agostino, M., Gabbay, D. M., Hähnle, R., & Posegga, J. (2013). *Handbook of Tableau Methods*. Springer Netherlands.

Dancy, J., & Celma, J. L. P. (1993). *Introducción a la epistemología contemporánea*. Tecnos.

de Wind, P. (2001). *Modal logic in coq*. Vrije Universiteit.

Dretske, F. (2005). Is Knowledge Closed Under Known Entailment? The Case Against Closure. En M. Steup & E. Sosa (Eds.), *Contemporary Debates in Epistemology* (pp. 13–26). Blackwell.

Enderton, H. B. (2001). *A Mathematical Introduction to Logic*. Harcourt / Academic Press.

Fitting, M. (2005). The logic of proofs, semantically. *Annals of Pure and Applied Logic*, *132*(1), 1–25.

Fitting, M. (2013). *TR–2013005: Realization Implemented* (Computer Science Technical Reports No. Paper 380). CUNY Ph.D. Program in Computer Science. Recuperado a partir de http: / / academicworks.cuny.edu / gc_cs_tr / 380

Gabbay, D. M., & Guenthner, F. (2010). *Handbook of Philosophical Logic: Volume 15* (2nd ed.). Springer Publishing Company, Incorporated.

Gettier, E. (1963). Is Justified True Belief Knowledge? *Analysis*, *23*(6), 121–123.

Hakli, R., & Negri, S. (2012). Does the deduction theorem fail for modal logic? *Synthese*, *187*(3), 849–867.

Halpern, J. Y., & Pucella, R. (2007). Dealing With Logical Omniscience: Expressiveness and Pragmatics. *CoRR*, *abs/cs/070*.

Hawthorn, J. (1990). Natural deduction in normal modal logic. *Notre Dame Journal of Formal Logic*, *31*(2), 263–273.

Hintikka, J. (1962). *Knowledge and Belief*. Ithaca, N.Y.,Cornell University Press.

Huth, M., & Ryan, M. (2004). *Logic in Computer Science: Modelling and Reasoning About Systems*. New York, NY, USA: Cambridge University Press.

Levesque, H. J. (1984). A Logic of Implicit and Explicit Belief. En R. J. Brachman (Ed.), *AAAI* (pp. 198–202). AAAI Press.

Lewis, C. I., & von Leibniz, G. W. F. (1918). *A Survey of Symbolic Logic*. University of California Press.

Pritchard, D. (2012). In defence of modest anti-luck epistemology. En K. Becker & T. Black (Eds.), *The Sensitivity Principle in Epistemology* (pp. 173–192). Cambridge University Press.

Pudlák, P. (1998). *The Lengths of Proofs*.

Russell, S. J., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (2a ed.). Pearson Education.

Shapiro, S. (1998). Logical Consequence: Models and Modality. En M. Schirn (Ed.), *The Philosophy of Mathematics Today* (pp. 131–156). Clarendon Press.

Sim, K. M. (1997). Epistemic logic and logical omniscience: A survey. *International Journal of Intelligent Systems*, *12*(1), 57–81.

Sørensen, M. H. B., & Urzyczyn, P. (1998). *Lectures on the Curry-Howard Isomorphism*.

Stalnaker, R. (1991). The Problem of Logical Omniscience, I. *Synthese*, *89*(3), pp. 425–440.

Troelstra, A. S., & Schwichtenberg, H. (2000). *Basic Proof Theory (2Nd Ed.)*. New York, NY, USA: Cambridge University Press.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 433–460.