ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE CUADROS

Parte I

LÓGICAS DE DESCRIPCIÓN

1 INTRODUCCIÓN

Cuando pensamos en el concepto de simetría, quizá lo primero que se nos viene a la cabeza puede ser un polígono regular, un número capicúa o incluso alguna imagen del Test de Rorschach (Figura 1.1).

Figure 1.1: Lámina 8 del Test de Rorschach

Entonces pensamos en algo que se puede *partir sobre un eje*, y repite el mismo patrón, aunque quizá con direcciones diferentes, en ambas partes. En la resolución de problemas matemáticos es muy usual recurrir a las simetrías en términos de razonamiento, ya que, si el problema es simétrico, se puede analizar un caso en detalle y dicho análsis sería igual para sus pares simétricos.

Esto se lo robé a Ezequiel

En el contexto de razonamiento automático, se puede definir como simetría de una fórmula a una permutación de sus variables (literales) que mantenga su estructura y sus modelos. De esta manera, si somos capaces de demostrar que una fórmula tiene simetrías, podemos guiar al algoritmo de búsqueda para que solo busque soluciones en las partes no simétricas del espacio de búsqueda.

1.1 QUÉ VAMOS A HACER EN ESTA TESIS

En esta tesis vamos a trabajar con un tipo de base de datos muy particulares, llamadas *bases de conocimiento*. Y como en cualquier base de datos, el objetivo es poder hacer consultas y obtener de la manera más rápida posible, una respuesta correcta. Nos concetraremos en buscar métodos para agilizar la forma en que se realizan las consultas a nuestra base de conocimientos. Más concretamente estos métodos tienen como objetivo encontrar simetrías en la base de datos, por lo que si, por ejemplo, queremos hacer una pregunta sobre algún concepto muy *complicado*, A, pero nuestro algoritmo encuentra que A

es simétrico a B, y ya que conocemos la respuesta para B, entonces podemos reutilizar dicha respuesta.

CONSULTA DE USUARIO

↓

ALGORITMO DE CONSULTA ←TRABAJAREMOS HACIENDO ESTA PARTE MÁS EFICIENTE

↓

BASE DE CONOCIMIENTO

1.2 POR QUÉ

1.3 ANTECEDENTES SIMILARES, LA ESTRUCTURA DE ESTA TESIS

Trabajos similares [8] han sido desarrollados para lógicas modales. Trabajo en el cual esta tesis se apoya fuertemente ya que usa herramientas implementadas para dicho proyecto.

1.4 SOBRE SIMETRÍAS EN AR

2

SOBRE DL

Antes de comenzar con el desarrollo de las herramientas y los resultados obtenidos, fijaremos bases dando una descripción del dominio, empezando con una descripción informal de las lógicas de descripción.

2.1 LÓGICAS DE DESCRIPCIÓN

Como sabemos la lógica de primer orden[10] es una lógica en la cual la satisfacibilidad es indecidible, cuando no se aplica ninguna restricción. Las lógicas de descripción son fragmentos decidibles de la lógica de primer orden.

Más precisamente, las lógicas de descripción son una familia de la formalización de la Representación de Conocimiento basado en lógica. La investigación en el área de la representación y razonamiento de conocimiento, es dirigida en métodos que sean capaces de proveer descripciones de alto nivel del mundo, que puedan ser usadas eficazmente para construir aplicaciones inteligentes. En este contexto, nos referimos a un sistema "inteligente" a la habilidad que tiene un sistema para encontrar consecuencias implícitas en base a su conocimiento represantado de forma explícita. Tales sistemas son caracterizados como sistemas basados en conocimiento.

Distintos abordajes para la representación del conocimiento se desarrollaron en los años setenta, y son clasificados en dos categorías: formalismos basados en lógica (que nacen la intuición que el cálculo de predicados puede usarse de forma objetiva para capturar hechos del mundo), y por otro lado, representaciones que no son basados en la lógica. Entre los sistemas más populares que no son basados en lógica encontramos dos: las *redes semánticas* y *estructuras (frames)*. En las redes semánticas [9] el conocimiento se representa en la forma de un grafo dirigido con etiquetas: cada nodo es asociado con un concepto y los arcos representan relaciones de algún tipo entre conceptos. En el caso de las estructuras[7], el objetivo se centra en lograr conocimiento estructurado basándose en la capacidad de expresar relaciones entre estructuras. Una estructura representa cierto concepto y es caracterizada por un númeo de atributos; donde los valores de los atributos pueden ser elementos de algún dominio concreto (e.g., enteros, *strings*) o identificadores de otras estructuras. Si bien las redes semánticas y las estructuras contienen diferencias muy marcadas, tanto como en las motivaciones como en sus prestaciones, comparten una base común. De hecho, nos podemos referir

a ambas como estructuras de red, donde la estructura de cada red apunta a representar conjuntos de individuos y cómo éstos se relacionan entre sí. Por ejemplo, la figura 2.1 representa conocimiento modelando una pequeña parte del mundo animal; la estructura utilizada también es referida como una *terminología*, y representa la generalización/especificación de los conceptos involucrados. Por ejemplo, el vínculo entre entre *Félidos* y *Carnívoros* indica que todos los félidos son carnívoros; este tipo de relación también es llamada una relación *IS-A* (en español, "es-un/una"). La relación IS-A define una jerarquía entre los conceptos que conecta, como también la herencia de atributos: si un concepto A es más específico que otro concepto B, entonces el concepto A hereda las propiedades del concepto B. Por ejemplo, si decimos que los carnívoros tienen órganos para capturar a su presa, entonces los félidos también cuentan con dichos órganos. Un atributo característico de las Lógicas de Descripción es poder representar otros tipos de relaciones que siguen siendo válidas entre conceptos, más allá de las relaciones IS-A. Por ejemplo, el concepto Carnívoro cuenta con la propiedad (también llamada *rol*) expresada conectando al concepto con el nodo para el rol llamado *seAlimentaDe*. Este rol cuenta con una *restrición de valor*, denotado por la etiqueta r/v, que expresa una limitación en el rango de tipos de objetos que pueden relacionarse con el rol. Nuestro rol tiene una restricción en cuanto a la cardinalidad, expresada como (0, NIL), donde el primer número es el límite inferior y el segundo número el límite superior (NIL representa infinito). Entonces, podría leerse como *"un carnívoro es un animal que ha comido otros animales"*.

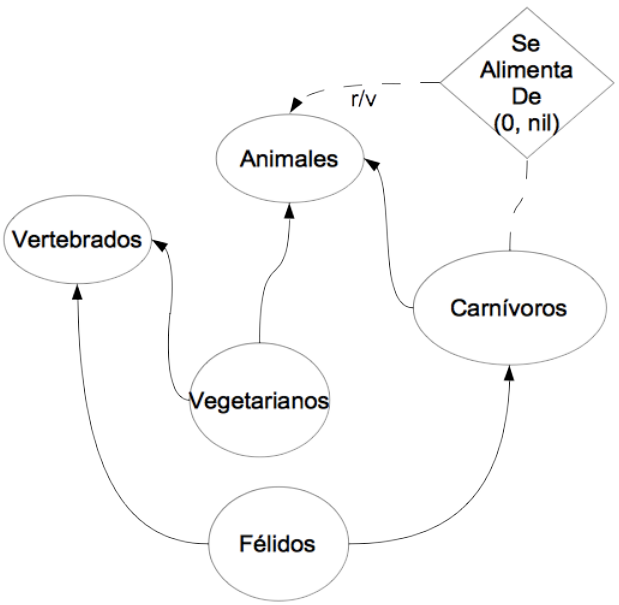


Figure 2.1: Red simple

Un detalle importante en estas estructuras son las relaciones implícitas que se encuentran entre los conceptos. Por ejemplo, como todos los carnívoros son animales, y un félido es un carnívoro, por lo tanto un félido es un animal. Y más que un detalle, este es el objetivo de un sistema de representación de conocimiento: encontrar este tipo de relaciones implícitas (y relaciones más complejas también).

Sin embargo, a medida que las relaciones entre los conceptos se vuelve más compleja, aumenta la dificultad para dar caracterizaciones precisas sobre qué tipos de relaciones pueden ser computadas, y para garantizar su correctitud y completitud.

Usando las ideas que hemos visto hasta el momento, varios sistemas fueron implementados y utilizados en muchas aplicaciones. Como resultado, surgió la necesidad de una caracterización precisa del significado de las estructuras usadas en las representaciones y del conjunto de inferencias que podían obtenerse de dichas estructuras. Esta precisa caracterización del significado de una red puede obtenerse simplemente definiendo un lenguaje para los elementos de la estructura (sintaxis) y proveyendo una interpretación para estos símbolos (semántica). Si bien la sintaxis puede variar bajo ciertos aspectos (dominio del problema, filosofía, etc), la semántica suele estar data por estilos de semántica de Tarski.

Para la sintaxis introducimos un lenguaje abstracto, bastante parecido a algún formalismo básico. La construcción básica se da a partir de dos alfabetos disjuntos que son utilizados para denotar *conceptos atómicos* (símbolos unarios de predicados) y *roles atómicos*, designados por símbolos de predicados binarios (utilizados para expresar relaciones entre los conceptos). Los términos, o fórmulas, son luego formados a partir de los símbolos básicos usando distintos constructores. Por ejemplo, la *intersección de conceptos*, que es denotada por $C \sqcap D$, es usada para restringir el conjunto de individuos bajo consideración a aquellos que pertenecen tanto a C como a D.

Las características fundamentales de las Lógicas de Descripción reside en las construcciones que permiten establecer relaciones entre los conceptos. Los más básicos son las *restricciones sobre valores*. Por ejemplo, una restricción de valor, escrita $\forall R.C$, requiere que todos los individuos que estén en la relación R bajo el concepto descrito, pertenezcan al concepto C (técnicamente, esto significa que todos los individuos que están en la relación R con un individuo descrito por el concepto en cuestión, pueden ser descritos como individuos de C).

Con respecto a la semántica, básicamente los conceptos tienen interpretaciones de conjuntos teóricos: un concepto es interpretado como un conjunto de individuos (elementos de ese conjunto) y los roles son interpretados como conjuntos de pares de individuos. Ahondaremos más en estos temás más adelante en este capítulo.

Por ende, los conceptos atómicos son interpretados como subconjuntos del dominio de interpretación, mientras que la semántica de

los otros constructores es especificada definiendo el conjunto de individuos denotado por cada constructor. Por ejemplo, el concepto $C \sqcap D$ es el conjunto de individuos obtenido intersecando los conjuntos de individuos denotados por C y por D, respectivamente. De manera similar, la interpretación de $\forall R.C$ es el conjunto de individuos que están en la relación R con los individuos que son parte del conjunto denotado por C.

A modo de ejemplificar un poco, supongamos que *Persona, Hombre* y *Mujer* son conceptos atómicos y que *tieneHijo* son roles atómicos. Usando los operadores *intersección, unión* y *complemento* de conceptos, interpretados como operaciones entre conjuntos, podemos describir nuevos conceptos, como por ejemplo "las personas que no son mujeres" o "los individuos que son mujeres u hombres", denotados por las siguientes expresiones

$$Persona \sqcap \neg Mujer \quad \text{y} \quad Mujer \sqcup Hombre$$

La inferencia básica sobre expresiones entre conceptos en Lógicas de Descripción es la *subsunción* (inclusión), escrita por lo general como $C \sqsubseteq D$. Determinar la inclusión refiere al problema de comprobar si el concepto denotado por D *(subsumidor)* es considerado más general que el concepto denotado por C. En otras palabras, comprueba si el concepto C *(subsumido)* siempre denota un subconjunto del conjunto denotado por D. Por ejemplo, uno podría querer saber si $Persona \sqsubseteq Mujer$. Otro tipo de inferencia sobre expresiones de conceptos es la de *satisfacibilidad*, que es el problema de comprobar si una expresión no denota necesariamente el concepto vacío. De hecho, la satisfacibilidad de conceptos es un caso especial de inclusión con el conjunto subsumidor siendo el conjunto vacío, significando que el concepto no es satisfacible.

Aunque ya hemos especificado el significado de los conceptos con una semántica lógica, los diseños de procedimientos de inferencia sobre Lógicas de Descripción fueron influenciados por mucho tiempo por las redes semánticas, donde los conceptos eran vistos como nodos y los roles como aristas en una red. La inclusión entre expresiones de conceptos fue reconocida como la inferencia clave y la idea básica de los primeros algoritmos de inclusión fue transformar dos conceptos, tomados como entrada, a grafos etiquetados y ver si alguno podía ser embebido en el otro, el grafo embebido correspondería al concepto más general (el subsumidor) [6].

Este método es llamado *comparación estructural*, y la relación que se computa entre conceptos es llamada *inclusión estructural*. Sin embargo, un análisis minucioso sobre el algoritmo de inclusión estructural muestra que es correcto, pero no siempre completo en término de la semántica lógica: siempre que el algoritmo retorna "sí", la respuesta es correcta, pero cuando retorna "no", existe la posibilidad que esa respuesta sea incorrecta.

La motivación para contar con algoritmos de inclusión que sean completos vienen de la mano de que en el uso de sistemas de representación de conocimiento es por lo general necesario contar con la garantía que el sistema no ha fallado en verificar inclusiones.

En el paper *"The Tractability of Subsumption in Frame-Based Description Logics"* [2], argumentan que hay una *compensación/pérdida* (o, *tradeoff*, en inglés) entre la expresividad de un lenguaje de representación y la dificultad de hacer razonamientos sobre las representaciones construidas usando dicho lenguaje. En otras palabras, mientras más expresivo es el lenguaje, más complicado es razonar sobre el mismo. También proveen un primer ejemplo de este tradeoff analizando el lenguaje $\mathcal{FL}^-$ (*Frame Language*), que incluye intersección de conceptos, restricción de valores y una simple forma de cuantificación existencial.

El mismo paper introduce al menos dos nuevas ideas:

- "eficiencia de razonamientos" sobre estructuras de conocimiento pueden ser estudiadas usando las herramientas de complejidad computacional;
- diferentes combinaciones de constructores pueden dar lugar a lenguajes con diferentes propiedades computacionales [1].

Una consecuencia inmediata que se desprende de esas ideas es que el tradeoff entre la complejidad computacional y la expresividad del lenguaje (que el mismo está definido en término de los constructores que están admitidos en el lenguaje) puede ser estudiada formal y metódicamente. Más aún, el problema de encontrar el tradeoff óptimo, siendo éste la extensión más expresiva de $\mathcal{FL}^-$ con respecto a un conjunto de constructores dados que mantiene las inclusiones en el orden polinomial, fue muy estudiadio [1].

2.1.1 *Representación del Conocimiento en Lógicas de Descripción*

En la sección previa, un lenguaje de representación básico para Lógicas de Descripción fue introducido junto con algunas técnicas de razonamiento claves. Ahora ilustraremos cómo las Lógicas de Descripción pueden ser útiles en el diseño de aplicaciones que se basan en el conocimiento, o sea, como un lenguaje $\mathcal{DL}$ (por *Description Logics* en inglés) es utilizado en un sistema de representación de conocimiento que provee un lenguaje para definir bases de conocimiento y herramientas para llevar a cabo inferencias sobre el mismo.

Realizar sistemas de conocimiento tiene dos aspectos claves:

- Proveer una caracterización precisa de la base de conocimientos; esto involucra caracterizar con precisión el tipo de conocimiento a ser especificado al sistema como también definir claramente

1 http://www.cs.man.ac.uk/~ezolin/dl/

los servicios de razonamiento que el sistema proveerá, o sea, el tipo de preguntas que el sistema debería ser capaz de responder.

- Proveer un entorno de desarrollo donde el usuario pueda dar uso de distintos servicios y que éstos sean eficientes.

Dentro de una base de conocimientos, uno es capaz de ver la marcada distinción entre el conocimiento *intencional*, conocimiento general acerca del dominio del problema, y *extensional*, específico a un problema en particular. Dichos conocimientos están capturados por dos componentes, la *TBOX* y la *ABOX*. La primera contiene conocimiento intencional en forma de una *terminología* (de ahí el término *Terminological Box*), y es construida a través de declaraciones que describen propiedades generales sobre los conceptos. Dada la naturaleza de las inclusiones las relaciones sobre los conceptos que constituyen la terminología, las Tbox son pensadas como estructuras de tipo reticulado.

La Abox, es la encargada del conocimiento extensional, o aserciones (de ahí *Assertional Box*). Este conocimiento es específico para individuos del dominio.

2.1.1.1 *TBox*

Un elemento clave en una base de conocimiento basada de Lógicas de Descripción ($\mathcal{DL}$, de ahora en más) está dado por las operaciones usadas para construir la terminología. Tales operaciones están relacionadas de manera directa a las formas y significados de las declaraciones permitidas en la TBox.

La forma más básica de declaración en una TBox es la *definición* de concepto. Por ejemplo, podríamos definir el concepto de mujer como las personas que no son hombres

$$Mujer \equiv Persona \sqcap \neg Hombre$$

Tal declaración es interpretada como una equivalencia lógica, que provee condiciones necesarias y suficientess para clasificar individuos como mujeres. Es importante notar que esta forma de definición es mucho más fuerte que otras utilizadas en otros tipos de representación de conocimiento, que típicamente solo proveen las condiciones necesarias; el punto fuerte de este tipo de declaraciones es usualmente considerado una característica particular de las bases de conocimiento de $\mathcal{DL}$.

Por lo tanto, en estas bases, la terminología es constituida por un conjunto de definiciones de conceptos. Sin embargo, hay un par de reglas básicas implícitas en cualquier terminología:

- Solo una definición por concepto es permitida

- Las definiciones son *acíclicas*, es decir, los conceptos nunca son definidos en términos de otros conceptos que directamente, o indirectamente, se refieren al concepto en cuestión. Como tampoco un concepto puede estar definido en términos de sí mismo.

En particular, la tarea principal de construir una terminología es la *clasificación*, que refiere a colocar una nueva expresión de concepto en su lugar correcto con respecto a la jerarquía de conceptos visto como taxonomía. La clasificación puede lograrse verificando las relaciones de inclusión entre todos los conceptos junto con la nueva expresión de conceptos. Las expresiones de tipo

$$Hombre \sqsubseteq Persona$$

son llamadas *axiomas de inclusión generales.*

2.1.1.2 *ABox*

Como mencionamos, la ABox contiene conocimiento extensional acerca del dominio de interés, i.e., aserciones acerca de los individuos. Veamos como ejemplo las siguientes aserciones

$$Mujer \sqcap Persona(Julieta)$$
$$tieneCapital(Islandia, Reikavik)$$

La primera indica que el individuo *Julieta* es una persona que es mujer, mientras que la segunda expresa que la capital de Islandia es Reikiavik. Las aserciones del primer ejemplo son usualmente llamadas *aserciones sobre conceptos*, mientras que las segundas son llamadas *aserciones sobre roles*.
La tarea de razonamiento básica en una ABox es *comprobación de instancias*, la cual verifica si algún usuario en particular es una instancia (o pertenece) a cierto concepto.
Es importante resaltar que la presencia de de individuos en una base de conocimientos aumenta la complejidad de razonamiento desde un punto computacional [4].

2.1.2 *Sintaxis y Semántica*

Habiendo ya fijado las bases conceptuales de un sistema de representación de conocimiento $\mathcal{DL}$ pasaremos a describir la sintaxis y la semántica de manera formal de algunos de estos sistemas. En la Tabla 2.1 podemos observar la sintaxis y semántica de una lógica muy simple llamada $\mathcal{AL}$ (por *Attribute Language*). En dicha tabla, A, C y D representan Conceptos y R representa un Rol atómico. La semántica

SINTAXIS	SEMÁNTICA	COMENTARIO
A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	Concepto Atómico
R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$	Rol Atómico
$\top$	$\Delta^{\mathcal{I}}$	Top, concepto más general
$\bot$	$\emptyset$	Bottom, concepto más específico
$\neg A$	$\Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}$	Negación de concepto atómico
$C \bigsqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	Intersección de Conceptos
$\forall R.C$	$\{a \in \Delta^{\mathcal{I}} \mid \forall b.(a,b) \in R^{\mathcal{I}} \Rightarrow b \in C^{\mathcal{I}}\}$	Restricción de valor
$\exists R.\top$	$\{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}}\}$	Cuantificación existencial

Table 2.1: Sintaxis y semántica de AL

es definida usando una función de interpretación $\mathcal{I}$, y un conjunto no vacío $\Delta^{\mathcal{I}}$ (el dominio de la interpretación), que asigna un conjunto $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ a cada concepto atómico A, y asigna una relación binaria $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ a cada Rol Atómico R.
En la Tabla 2.2 se extiende la función de interpretación $\mathcal{I}$ para dar lugar a más constructores y así enriquecer la expresividad.

¿AGREGAR EJEMPLOS?

<Ejemplos>

2.1.3 *Clases y Complejedidad*

Como fue mencionado anteriormente, a medida que agregamos constructores para poder generar nuevas expresiones y enriquecer la expresividad de nuestro lenguaje, el costo computacional aumenta. En la Tabla 2.3 listamos algunos lenguajes junto a su complejidad computacional[2].

2.2 SU RELACIÓN CON LÓGICAS MODALES

Las principales herramientas para realizar los experimentos sobre Lógicas de Descripción fueron desarrolladas principalmente para Lógicas Modales, un fragmento particular (y decidible) de Lógica de Primer Orden.

2 http://www.cs.man.ac.uk/~ezolin/dl/

NOMBRE	SINTAXIS	SEMÁNTICA	COMENTARIO
$\mathcal{U}$	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	Unión de dos Conceptos
$\mathcal{E}$	$\exists R.C$	$\{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}} \wedge b \in C\}$	Cuantificación Completa
$\mathcal{N}$	$\geq nR$ $\leq nR$	$\{a \in \Delta^{\mathcal{I}} \mid\mid \{b.(a,b) \in R^{\mathcal{I}} \mid \geq n\}$ $\{a \in \Delta^{\mathcal{I}} \mid\mid \{b.(a,b) \in R^{\mathcal{I}} \mid \leq n\}$	Restricción de cardinalidad
$\mathcal{C}$	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	Negación de un concepto Arbitrario

Table 2.2: Extensión de AL

$\mathcal{DL}$	COMPLEJIDAD COMPUTACIONAL DE INCLUSIONES
$\mathcal{AL}$	*PTime*
$\mathcal{ALC}$	*PSpace*
$\mathcal{SHIF}$	*ExpTime*
$\mathcal{SHOIN}$	*NExpTime*

Table 2.3: Lenguajes $\mathcal{DL}$ y su complejidad

A continuación, una breve introducción sobre Lógicas Modales y luego su relación con Lógicas de Descripción.

2.2.1 *Lógicas Modales*

DEFINICIÓN:

(Sintaxis) Sea $\mathsf{PROP} = \{p_1, p_2, ...\}$ *un conjunto contable infinito de variables proposicionales y* $\mathsf{MOD} = \{m, m''\}$ *un conjunto de símbolos de modalidades. Denominamos como* signatura modal *al par* $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{MOD} \rangle$. *El conjunto de las fórmulas modales básicas* FORM *sobre la signatura* $\mathcal{S}$ *se define como*

$$\mathsf{FORM} ::= p \mid \neg\varphi \mid \varphi \vee \psi \mid [m]\varphi,$$

donde $p \in \mathsf{PROP}$, $m \in \mathsf{MOD}$, *y* $\varphi, \psi \in \mathsf{FORM}$.$\top$ *y* $\bot$ *representan una tautología arbitraria y la contradicción, respectivamente. También utilizaremos los conectivos clásicos como* $\wedge, \rightarrow$ *definidos de la forma usual. Por cada* $m \in \mathsf{MOD}$ *tenemos un* operador dual $\langle m \rangle$ *(diamante) definido como* $\langle m \rangle \varphi = \neg[m]\neg\varphi$. *Cuando* MOD *es un singleton, es decir, en el caso monomodal, simplemente escribimos* $\Box$ *y* $\Diamond$.

DEFINICIÓN:

(Modelos) Un modelo (o modelo de Kripke) $\mathcal{M}$ *es una tripla* $\mathcal{M} = \langle W, \{R^m\}_{m \in \mathsf{MOD}}, V \rangle$, *donde:*

- *W, el* dominio, *es un conjunto no vacío. Los elementos de W se denominan* puntos, estados, mundos, *etc.*
- *Cada* R^m *es una relación binaria sobre W.*
- *V, la* valuación, *es una función que asigna a cada elemento* $p \in \mathsf{PROP}$ *un subconjunto* $V(p) \subseteq W$. *Informalmente, podemos pensar a* $V(p)$ *como el conjunto de variables proposicionales que son verdaderas en p.*

DEFINICIÓN:

(Semántica) Sea $\mathcal{M} = \langle R, M, V \rangle$ *un modelo y sea* φ *una fórmula Modal. Definamos la noción de que* φ *sea* satisfacible *en un modelo* $\mathcal{M}$ *en un punto* $w \in W$ *de la siguiente manera:*

$$\begin{aligned}
&\mathcal{M}, w \models p && \textit{sii} && w \in V(p) \textit{ para } p \in \mathsf{PROP},\\
&\mathcal{M}, w \models \neg\varphi && \textit{sii} && \mathcal{M}, w \not\models \varphi,\\
&\mathcal{M}, w \models \varphi \vee \psi && \textit{sii} && \mathcal{M}, w \models \varphi \text{ o } \mathcal{M}, w \models \psi,\\
&\mathcal{M}, w \models \Box\varphi && \textit{sii} && \mathcal{M}, v \models \varphi, \textit{ para todo } v \textit{ tal que } wRv.
\end{aligned}$$

DEFINICIÓN:

(Literales y CNF modal). Un literal proposicional *l es una variable proposicional p o su negación* $\neg p$. *El conjunto de literales sobre* PROP *es* $\mathsf{PLIT} = \mathsf{PROP} \cup \{\neg p \mid p \in \mathsf{PROP}\}$. *Una fórmula modal está en* forma normal conjuntiva modal (CNF modal) *si es una conjunción de cláusulas en CNF modal. Una cláusula* en CNF modal *es una disyunción de literales proposicionales y modales. Un* literal modal *es una fórmula de la forma* $[m]C$ *o* $\neg[m]C$, *donde C es una cláusula en CNF modal.*

DEFINICIÓN:

(Modelos) Un modelo (o modelo de Kripke) $\mathcal{M}$ *es una tripla* $\mathcal{M} = \langle W, \{R^m\}_{m \in \mathsf{MOD}}, V\rangle$, *donde:*

- *W, el* dominio, *es un conjunto no vacío. Los elementos de W se denominan* puntos, estados, mundos, *etc.*
- *Cada* R^m *es una relación binaria sobre W.*
- *V, la* valuación, *es una función que asigna a cada elemento* $w \in W$ *un subconjunto* $V(w) \subseteq \mathsf{PROP}$. *Informalmente, podemos pensar a* $V(w)$ *como el conjunto de variables proposicionales que son verdaderas en w.*

DEFINICIÓN:

(Modelos Puntuados) Un modelo puntuado *es un modelo con un elemento distinguido, e.g., dado un modelo* $\mathcal{M} = \langle W, R, V\rangle$ *y un elemento w en W, el modelo puntuado correspondiente es la tupla* $\mathcal{M} = \langle w, W, R, V\rangle$. *Si un modelo puntuado* $\mathcal{M}$ *satisface una fórmula* φ *escribimos* $\mathcal{M} \models \varphi$.

DEFINICIÓN:

(Semántica) Sea $\mathcal{M} = \langle w, W, R, V\rangle$ *un modelo puntuado y* φ *una fórmula en CNF modal. Definamos recursivamente cuándo una fórmula* φ *es* satisfacible *en* $\mathcal{M}$ *de la siguiente manera:*

$\mathcal{M} \models \varphi$	*sii*	*para todas las cláusulas* $C \in \varphi$, $\mathcal{M} \models C$,
$\mathcal{M} \models C$	*sii*	*existe un literal* $l \in C$ *tal que* $\mathcal{M} \models l$,
$\mathcal{M} \models p$	*sii*	$p \in V(w)$ *para* $p \in \mathsf{PROP}$,
$\mathcal{M} \models \neg p$	*sii*	$p \notin V(w)$ *para* $p \in \mathsf{PROP}$,
$\mathcal{M} \models \Box C$	*sii*	$\langle v, W, R, V\rangle \models C$, *para todo v tal que wRv,*
$\mathcal{M} \models \Box \neg C$	*sii*	$\mathcal{M} \not\models \Box C$

2.3 HACIA DL

Ahora trazaremos el nexo entre Lógicas Modales, $\mathcal{ML}$, y $\mathcal{DL}$. Como mencionamos anteriormente, distintas clases de lenguajes en $\mathcal{DL}$ cuentan con diferente expresividad, y por lo tanto, complejidad computacional. Algunos ejemplos son $\mathcal{EL}$, $\mathcal{ALC}$ y $\mathcal{SHOIQ}$, ordenados de menos expresivos a más expresivos. Por ahora nos concetraremos en $\mathcal{ALC}$.

2.3.1 *Nexo Sintáctico*

Las expresiones entre conceptos en $\mathcal{ALC}$ podrían pensarse como fórmulas modales en $\mathcal{ML}$, como muestra la tabla 2.4

$\mathcal{ML}$	$\mathcal{DL}$
p_i	C_i, y C_i es un concepto
$\neg\varphi$	$\neg C_j$, y C_j es un concepto
$\varphi \vee \psi$	$C_j \sqcup C_i$, con Cj, C_i conceptos
$\varphi \wedge \psi$	$C_j \sqcap C_i$, con Cj, C_i conceptos
$\langle R\rangle\varphi$	$\exists R.C_j$, y R es un rol y C_j es un concepto
$[R]\varphi$	$\forall R.C_j$, y R es un rol y C_j es un concepto

Table 2.4: Relación entre $\mathcal{ML}$ y $\mathcal{DL}$

Quedémonos ahora con una Lógica Modal básica con los siguientes constructores: $p_i, \varphi \vee \psi, \langle R\rangle$ y de acuerdo a la tabla 2.4 escribiremos

$$\begin{array}{lll} C_i & \textit{en vez de} & p_i \\ \varphi \sqcup \psi & \textit{en vez de} & \varphi \vee \psi \\ \exists R.\varphi & \textit{en vez de} & \langle R\rangle\varphi \end{array}$$

2.3.2 *Nexo Semántico*

Tomaremos los modelos de $\mathcal{ML}$, o sea, grafos dirigidos y etiquetados.

2.3.2.1 *Extensión de una fórmula φ*

($\mathcal{ML}$) Sea $\mathcal{M} = \langle \mathcal{I}, \Delta \rangle$ un modelo de $\mathcal{ML}$, analizaremos la extensión de una fórmula φ ($\| \varphi \|^{\mathcal{M}}$) recursivamente de la siguiente manera:

- $\| p \|^{\mathcal{M}} = \{w \mid w \in \mathcal{I}(p)\}$, o sea, $\| p \|^{\mathcal{M}} = \mathcal{I}(p)$
- $\| \neg\varphi \|^{\mathcal{M}} = \Delta \setminus \| \varphi \|^{\mathcal{M}}$
- $\| \varphi \vee \psi \|^{\mathcal{M}} = \| \varphi \|^{\mathcal{M}} \cup \| \psi \|^{\mathcal{M}}$
- $\| \langle R \rangle \varphi \|^{\mathcal{M}} = \{w \mid \exists v. R(w, v) \wedge v \in \| \varphi \|^{\mathcal{M}}\}$

($\mathcal{DL}$) Sea $\mathcal{M} = \langle \Delta, \mathcal{I} \rangle$, con Δ un conjunto no vacío e $\mathcal{I}$ una función de interpretación, decimos que:

$$
\begin{array}{llll}
\langle \Delta, \mathcal{I} \rangle \models & C_i \sqsubseteq C_j & sii & \| C_i \|^{\mathcal{M}} \subseteq \| C_j \|^{\mathcal{M}} \\
\langle \Delta, \mathcal{I} \rangle \models & C_i \equiv C_j & sii & \| C_i \|^{\mathcal{M}} = \| C_j \|^{\mathcal{M}} \\
\langle \Delta, \mathcal{I} \rangle \models & a : C_i & sii & \mathcal{I}(a) \in \| C_i \|^{\mathcal{M}} \\
\langle \Delta, \mathcal{I} \rangle \models & (a, b) : R & sii & (\mathcal{I}(a), \mathcal{I}(a)) \in \mathcal{I}(R)
\end{array}
$$

Sea $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$ una Base de Conocomiento, o *Knowledge Base*, con $\mathcal{T}$ una TBox y $\mathcal{A}$ una ABox. Sea $\mathcal{M} = \langle \mathcal{I}, \Delta \rangle$ un modelo, definiremos $\mathcal{M} \models \mathcal{KB}$ si y solo si:

$$
\begin{array}{lll}
\forall C_i \sqsubseteq C_j \in \mathcal{T} & vale & \mathcal{M} \models C_i \sqsubseteq C_j \\
\forall a : C_i \in \mathcal{T} & vale & \mathcal{M} \models a : C_i \\
\forall (a, b) : R \in \mathcal{A} & vale & \mathcal{M} \models (a, b) : R
\end{array}
$$

Por último, ya estamos en condiciones de definir cuándo una expresión en $\mathcal{DL}$ es consecuencia lógica de una $\mathcal{KB}$:

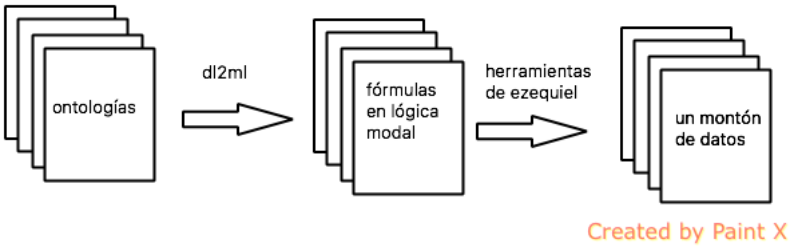
$$
\begin{array}{lll}
\mathcal{KB} \models C_i \sqsubseteq C_j & sii & \forall \mathcal{M}. (\mathcal{M} \models \mathcal{KB} \implies \mathcal{M} \models C_i \sqsubseteq C_j). \\
\mathcal{KB} \models a : C & sii & \forall \mathcal{M}. (\mathcal{M} \models \mathcal{KB} \implies \mathcal{M} \models a : C). \\
\mathcal{KB} \models (a, b) : R & sii & \forall \mathcal{M}. (\mathcal{M} \models \mathcal{KB} \implies \mathcal{M} \models (a, b) : R).
\end{array}
$$

3

SIMETRÍAS EN ONTOLOGÍAS

Ahora que sabemos que los modelos de Lógica Modal son los mismos que los de Lógicas de Descripción, podemos aplicar las mismas herramientas a ambas lógicas. Como las herramientas principales están desarrolladas para Lógica Modal, daremos traducciones de fórmulas en lógica de descripción a fórmulas en lógicas modales.

La arquitectura del sistema a granularidad baja es como sigue:



3.1 ONTOLOGÍAS

La historia de la inteligencia artificial nos ha mostrado que el *conocimiento* es crítico a la hora de desarrollar sistemas de inteligencia artificial. En muchos casos, a la hora de resolver alguna tarea, contar con *mejor* conocimiento puede llegar a ser más importante que contar con mejores algoritmos. Para contar con sistema verdaderamente inteligentes, el conocimiento necesita ser capturado, procesado, reusado y comunicado. Las ontologías dan soporte a todas estas tareas.

El término *ontología* puede ser definido como una especificación explícita de una *conceptualización*. La conceptualización describe conocimiento acerca del dominio, no sobre estados particulares del mismo; son especificadas o descritas usando algún lenguaje particular de modelado y términos particulares. La especificación formal de una conceptualización, i.e. la ontología, es requerida para poder dar soporte a las tareas antes mencionadas.

En conclusión, una ontología describe un dominio, mientras que una base de conocimiento (basada en una ontología) describe estados particulares de sus partes.

3.1.1 *Raíces Filosóficas*

El término *ontología* fue tomado prestado de la filosofía. Según el diccionario de Webster[1] una ontología es:

1 https://www.merriam-webster.com/dictionary/ontology

1. *Una rama de la metafísica relacionada con la naturaleza y las relaciones del ser.*

2. *Una teoría particular acerca de la naturaleza del ser, o los tipos de existencia.*

"Ontología", la *ciencia del ser*, es una palabra, tal como *metafísica*, que es usada bajo muchos contextos. A veces es considerada como idéntica a metafísica, pero aquí preferimos adoptarla en un sentido mucho más específico, como una parte de la metafísica; en particular, la parte que especifica las categorías más fundamentales de la existencia, las estructuras elementales de las que el mundo está construido. Por ende, *ontología*, será la palabra que utilizaremos en esta tesis para analizar los conceptos más generales y abstractos o distinciones que subyacen cada descripción más específica de cualquier fenómeno observable, e.g., tiempo, espacio, materia, proceso, causa y efecto, sistema, etc.

Recientemente, el término de *ontología* fue adoptado por investigadores en el área de la Inteligencia Artificial, quienes lo usan para designar los cimientos para los modelos del mundo. Un *agente* (e.g., un robot autónomo) usando un modelo particular solo podrá percibir *esa* parte del mundo que la ontología es capaz de representar. En este sentido, solo las cosas de su ontología son las que pueden existir para éste. De esta manera, una ontología se convierte en el nivel básico de un esquema de representación de conocimiento.

La Ontología, vista como *ciencia*, es una rama de la filosofía que lidia la naturaleza y la organización de la realidad. Intenta poder responder cosas como *qué es la existencia*, *cuáles son las propiedades pueden explicar la existencia*, etc. Aristóteles definió FALTA LA CITA a la ontología como la ciencia del ser. A diferencia de las ciencias *especiales*, en las que cada una investiga una clase particular de *seres* y sus determinaciones, la ontología apunta a *todas las especies que son y los atributos que le corresponden para ser.* En este sentido, la ontología *filosofal* intenta ser capaz de responder a la pregunta *¿qué es el ser?* o, reformulando de manera más significativa *¿cuáles son los atributos o características comunes a todos los seres?*.

FALTA CITA

"all the species qua being and the attributes that belong to it qua being"

Esto es lo que podría llamarse una *ontología general*, en contraste a los varios tipos de ontologías especiales que se refieren a algún dominio en particular. A dicha ontología, no le concierne la existencia de ciertos objetos, pero sí en cambio, una descripción rigurosa de sus características estructurales. En la práctica, la ontología puede pensarse como la teoría de la distinción, que puede ser aplicada independientemente al estado del mundo. En particular, nos interesamos por las distinciones:

1. Sobre las entidades del mundo (objetos físicos, eventos, religiones, etc)

2. Sobre las categorías teóricas utilizadas para modelar el mundo (conceptos, propiedades, calidad, estado, rol, parte, etc)

La investigación de la ontología en este sentido netamente filosófico es relevante para los sistemas basados en conocimiento. La ontología conforma una base para el conocimiento usado en estos sistemas.

3.1.2 *Qué es una Ontología*

Aunque la necesidad por contar con una definición formal de lo que es una ontología, todavía no hay una definición en común para dicho término, tal como sucede con el término *computabilidad efectiva*[3]. La definición puede ser categorizada dentro de tres grupos:

1. *Ontología* es un término utilizado en filosofía y su significado es *teoría de la existencia*.
2. *Ontología* es una especificación explícita de una conceptualización.
3. *Ontología* es un *cuerpo de conocimiento* que describe algún dominio, típicamente dominio del conocimiento de sentido común.

La primer definición es en el sentido filosófico que hemos discutido antes, sin embargo, tiene muchas implicaciones en los propósitos de Inteligencia Artificial (*IA*). La segunda definición es generalmente la adoptada por la comunidad de *IA*. Y la última definición interpreta a la ontología como el cuerpo interno del conocimiento, no como una manera para describirlo.

3.1.3 *Especificación de una Conceptualización*

La segunda definición de ontología mencionada en la sección anterior, *especificación explícita de una conceptualización*, fue dada por Thomas Gruber [5]. El significado exacto depende de lo que se entienda por *especificación* y *conceptualización*. Especificación explícita de una conceptualización significa que una ontología es una descripción (como una especificación formal de un programa) de los conceptos y relaciones que pueden existir para un agente o una comunidad de agentes. Esta definición es consistente con el uso de una ontología como un conjunto de definiciones de conceptos, pero de manera más general.

Una conceptualización puede definirse como una estructura de semántica intencional que codifica conocimiento implícito, restringiendo la estructura a una parte de un dominio. Una ontología es una especificación (parcial) de esta estructura, i.e., usualmente es una teoría lógica que expresa la conceptualización explícita en algún lenguaje. La conceptualización no depende del lenguaje, mientras que la ontología sí. El uso está ilustrado en la figura 3.1 ; muestra cómo una ontología restringe (i.e., define) el posible uso de constructores utilizados en la descripción del dominio.

La representación de una $\mathcal{KB}$ está basada en la especificación de la conceptualización. Una conceptualización es una vista simplifi-

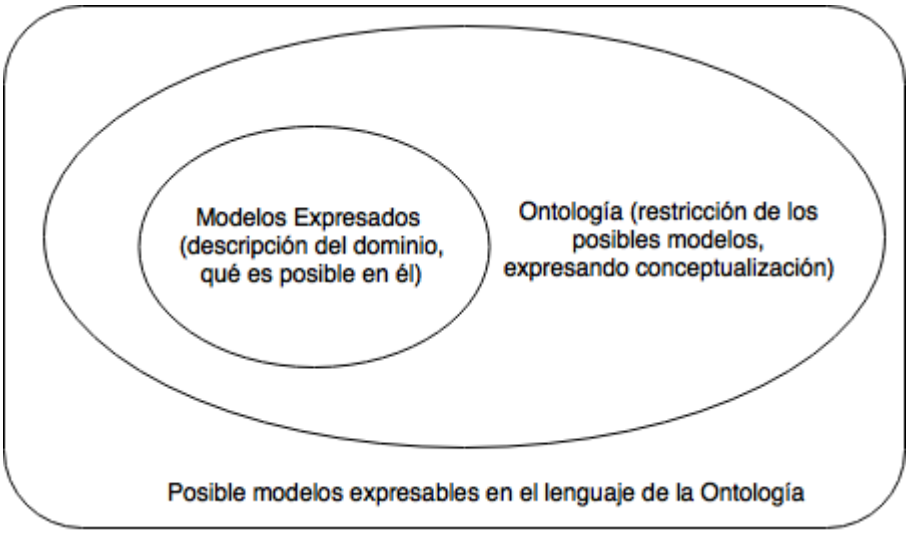


Figure 3.1: Una ontología expresando modelos para la descripción del dominio

cada y abstracta del mundo que deseamos representar para nuestro propósito. Cada $\mathcal{KB}$, sistema basado en conocimiento o agente está relacionado a alguna conceptualización, de forma explícita o implícita. Para estos sistemas, todo aquello que puede ser representado, es todo lo que *existe*. Cuando el conocimiento de un dominio es representado en algún formalismo declarativo, el conjunto de objetos que pueden ser representados se denomina *universo del discurso*. Este conjunto de objetos y las relaciones que se pueden describir entre ellos son reflejadas en el vocabulario de representación con el que un programa basado en conocimiento representa al conocimiento. Por ende, en el contexto de la *IA*, podemos describir la ontología de un programa definiendo un conjunto de términos de representación. En dicha ontología, las definiciones asocian los nombres de las entidades del universo en cuestión (e.g., clases, relaciones, funciones, u otros objetos) con descripciones que los nombres definen, y axiomas formales que restringen la interpretación y el uso correcto de estos términos. Formalmente, podría decirse que una ontología es una declaración de una teoría lógica.

FEO: CAMBIAR

El punto fuerte de las ontologías es por lo general la *taxonomía*. La taxonomía es la clasificación de cosas en forma jerárquica. Es por lo general vista como un árbol o un reticulado que expresa la relación de *subsunción*, i.e., si A subsume a B significa que todo lo que pertence a A también pertenece a B (i.e., la subsunción es la relación de inclusión).

De todas maneras, las ontologías no deben ser limitadas a jerarquías de taxonomías de clases ni a definiciones que solo introducen terminilogías y que no aportan ningún tipo de conocimiento acerca de ese mundo. Para especificar una conceptualización, los axiomas que restringen las posibles interpretaciones para los términos definidos,

también pueden llegar a ser necesarios. En forma pragmática, una ontología define el vocabulario con el cual se pueden generar consultas y aserciones que son intercambiadas entre los agentes.

3.1.4 *El Cuerpo del Conocimiento*

A veces, la ontología es definida como un cuerpo de conocimiento describiendo cierto dominio, típicamente un dominio de conocimiento de sentido común, usando un vocabulario de representación. En este caso, la ontología no es solo el vocabulario, sino la base de conocimiento (incluído el vocabulario que es usado para definir tal base de conocimiento).

El ejemplo típico es el proyecto *CYC* [2] que define su base de conocimiento como una ontología para cualquier otro sistema basado en conocimiento. CYC es un intento para realizar *IA* simbólica en una escala masiva capturando conocimiento común que es requerido para hacer tareas, triviales para una persona, pero muy difícil para una computadora. Todo el conocimiento en CYC es representado declarativamente en forma de aserciones lógicas. CYC contiene más de 400.000 aserciones principales, que incluyen desde simple declaraciones sobre hechos, reglas sobre qué conclusiones obtener si ciertos hechos son verdaderos, y reglas sobre cómo razonar con cierto tipo de hechos y reglas. También es capaz de obtener nuevas conclusiones a través de su motor de inferencias basándose en razonamiento deductivo.

En un sentido general, no es posible distinguir de forma objetiva qué debería pertener a la ontología y qué debería pertencer en la base de conocimiento. Depende de los requerimientos o de la naturaleza del uso de dicha ontología.

3.2 EXPRESANDO ONTOLOGÍAS

De ahora en más, nos referiremos a las ontologías como una *especificación explícita de una conceptualización*. En este sentido, la ontología restringe el uso previsto de los términos en ontología (ilustrado en 3.1) para que la ontología constituya un vocabulario y axiomas que pueden ser usados para expresar una base de conocimiento y puede ser usada para compartir conocimiento entre diferentes sistemas.

También requerimos que la especificación sea formal (como un programa que es escrito formalmente en algún lenguaje de programación), así de esta manera la ontología puede ser procesada por una computadora. A continuación describiremos algunos de estos lenguajes para expresar ontologias de manera formal.

2 http://www.cyc.com/

3.2.1 *Representación Formal*

Las ontologías, si son utilizadas para procesamiento automático en computadoras, necesitan ser especificadas de manera formal. Hay varios lenguajes para expresar ontologías de manera formal, revisaremos algunos de ellos a continuación. Estos lenguajes proveen métodos para expresar una ontología particular, y por lo general, también pueden utilzarse para expresar conocimiento base basado en la ontología.

La formalidad de la descripción de ontologías está resumida en la figura 3.2. En la parte derecha de la escala hay términos catalogados utilizados para expresar conocimiento o información. Estos términos pueden no con descripción alguna, y son comprendidos solo porque fueron elegidos del lenguaje natural, y su significado puede ser solo estimado. La descripción de cada término en lenguaje natural es mejor, especialmente si hay relaciones expresadas entre tales términos, como *es-un*, *parte-de*, *relacionado-con*, etc. Sin embargo, hasta que esta descripción está en lenguaje natural, la cual no está definida de manera forma, no solemos llamar *Ontología* a tal especificación.

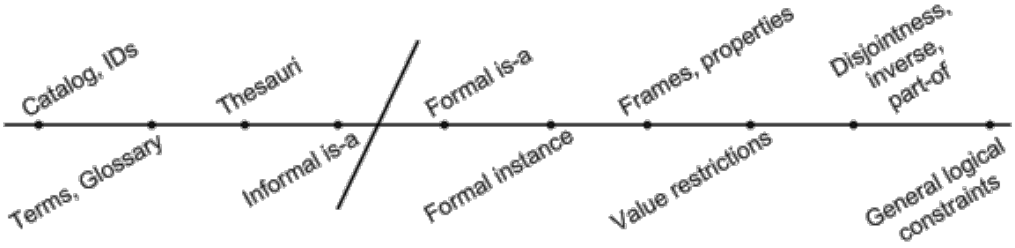


Figure 3.2: Levels of expressivity in ontology description

El punto sobre especificar formalmente una conceptualización, es cuando al menos una relación formal es definida y utilizada entre términos. Típicamente esto es relaciones formales *es-un*, taxonomias de clases y propiedades, relaciones de instanciación de clases, etc. Una relación formal se refiere a que es expresada en un lenguaje formal y que es posible concluir implicaciones que se desprenden del hecho que dos términos están conectados por la relación. En un nivel más alto, existen lenguajes que proveen la maquinaria necesaria para describir ontología, como *estructuras* o lógicas más simples como *lógicas de descripción*. Estos lenguajes permiten describir clases, instancias, y relaciones entre ellos, y también restricciones sobre su uso. En la parte de la derecha de la escala hay una teoría de lógica general, donde es posible usar lógicas completas (como lógica de primer orden, lógica modal, etc) para expresar el uso de los términos que son usados en ontología.

VER POR QUÉ PONE ONTOLOGY SOLO

En la escala presentada en la figura 3.2 también se expresa la usabilidad de las ontologías. La parte descrita informalmente no es de mucho uso a la hora de procesamiento automático. Por supuesto

que mientras se use una descripción más informal, más fácil va a ser desarrollar una ontología. Por otro lado, si utilizamos descripciones más formales y expresivas, tales descripciones pueden ser procesadas de forma automática, podemos capturar de una mejor manera el significado pretendido, y a la hora de compartir una ontología la tarea se vuelve más amena. Cuando trabajamos con una ontología, por lo general deseamos mantener cierta consistencia. Sin embargo, no siempre es posible mantener un sistema completo y decidible. Por ejemplo, cuando utilizamos descripciones en predicados de lógica de primer orden, no podemos garantizar decibilidad [10], pero puede ser garantizada en lógicas de descripción menos expresivas. Por lo tanto, en casos prácticos se busca un balance entre expresividad y propiedades computacionales de la descripción.

3.2.2 *Formalismos para modelar ontologías*

A continuación listaremos algunos lenguajes muy utilizados para modelar ontologías, no daremos descripciones de éstos ya que no son del interés para esta tesis, salvo, por supuesto, las lógicas de descripción, que fueron descritas en el capítulo 2:

- *Modelos basados en estructuras*
- *Redes semánticas*
- *Grafos conceptuales*
- *Formato de intercambio de conocimiento*
- *Lógicas comunes*
- *Lógicas de descripción*

3.3 LA WEB SEMÁNTICA

La *Worl Wide Web*, o *WWW*, es una librería enorme de documentos entrelazados que son transferidos entre computadoras y presentados a las personas. Actualmente la WWW contiene muchísima información y conocimiento, pero las máquinas por lo general solo transportan y presentan el contenido de los documentos que describen al conocimiento. Pero son las personas las encargadas de conectar todas las fuentes de información relevante e interpretarlas.

La *Web Semántica* [3], tiene como objetivo mejorar la actual web para que las computadoras sean capaces de procesar la información presentada en la WWW, interpretarla y conectarla, para ayudar a las personas a encontrar el conocimiento requerido. De la misma manera

3 https://www.w3.org/2001/sw/

que la WWW, la web semántica es un sistema distribuido de hipertexto, está pensado para formar un gran sistema de conocimiento distribuido. El foco de la web semántica es compartir datos, en vez de documentos. En otras palabras, es un proyecto que debería proveer [4] *una estructura, o marco, común que permita a la información ser compartida y reusada a través de los límites de la aplicación, el trabajo y la comunidad. Es un esfuerzo en colaboración liderado por el Consorcio de la WWW (W3C).*

3.3.1 *Arquitectura*

La arquitectura de la web semántica está dada por la figura 3.3, solo describiremos algunos de estos componentes, aquellos que son relevantes para esta tesis.

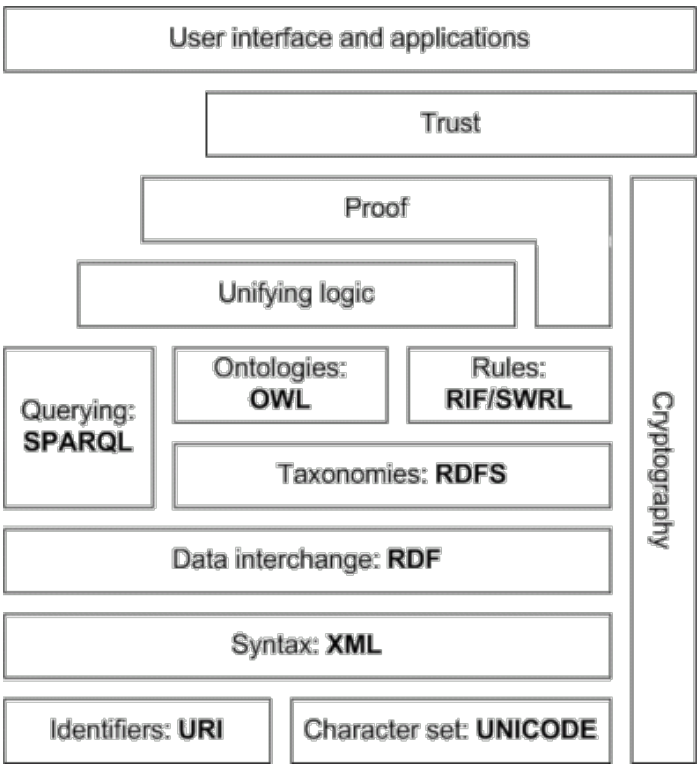


Figure 3.3: Web Semántica

La primer cada del sistema, *URI* y *Unicode*, siguen las características importantes de la WWW. Unicode es un estandar para la codificación de conjuntos internacionales de caractéres y permite usar cualquier lenguaje humano (para escritura y lectura) en la web usando una forma estandar. URI, o *Uniform Resource Identifier* (Identificador de recurso uniforme, en español) es un *string* (conjunto de caractéres) de forma estandarizada que permite identificar recursos de forma única (e.g., documentos). Toda variante internacionalizada de URI es *Internationalized Resource Identifier* (IRI) que permite el uso

4 https://www.w3.org/2001/sw/

de caractéres Unicode en identificadores, por lo que luego un *mapeo* a URI es definido.

La capa *Extensible Markup Language* (XML) junto con las definiciones de *XML namespace* y *XML schema* se aseguran que hay una sintaxis común utilizada en la web semántica. XML es un lenguaje de marcas (etiquetas) de propósito general para documentos que contienen información estructurada.

Un formato de representación de la información básico para la web semántica es *RDF* (por *Resource Description Framework*, o *Sistema de Descripción de Recursos*, en español). Es un sistema para representar información acerca de los recursos en forma de grafo. Está basado en triplas *sujeto-predicado-objeto* que conforma el grafo de datos. Todos los datos en la web semántica usan RDF como el principal lenguaje de representación.

Cualquiera puede un vocabulario de términos utilizados para descripciones más detalladas. Para permitir descripciones estandarizadas de taxonomías y otros constructores *ontológicos*, un *Esquema RDF* (*RDFS*) fue creado junto a su semántica formal dentro de RDF. RDFS pueden ser utilizados para describir taxonomías de clases y propiedades y así crear ontologías pequeñas.

Para crear ontologías más detalladas se puede utilizar el *Lenguaje Ontológico Web* (*OWL*, por *Ontology Web language*). OWL es un lenguaje derivado de las lógicas de descripción y ofrece más construcciones sobre RDFS. Está sintácticamente embebido dentro de RDF, así que como RDFS, provee vocabulario estandarizado adicional. Dado que OWL está basado en $\mathcal{DL}$, no es de extrañar que haya una defínción formal de este lenguaje.

3.3.2 *Resource Description Framework*

Como se mencionó anteriormente RDF es un sistema para representar información acerca de recursos en forma de grafo. Debido a que estaba pensando principalmente para representar metadatos sobre recursos de la WWW, fue construido con recursos de URI.

Veamos un ejemplo, expresemos lo siguiente: *Argentina tiene-capital BuenosAires*. La tripla que describe esta información está representada por la figura 3.4. Todos los elementos de esta tripla son recursos definidos por URI. El primer recurso, *http://www.famaf.unc.edu.ar/ejemplo.rdf#Argentina* tiene como propósito identificar a Argentina (el sujeto). El segundo recurso *http://www.famaf.unc.edu.ar/ejemplo/relaciones#tieneCapital* (predicado) es el predicado de tener capital. El último recurso es la capital de Argentina *http://www.famaf.unc.edu.ar/ejemplo.rdf#BuenosAires.*

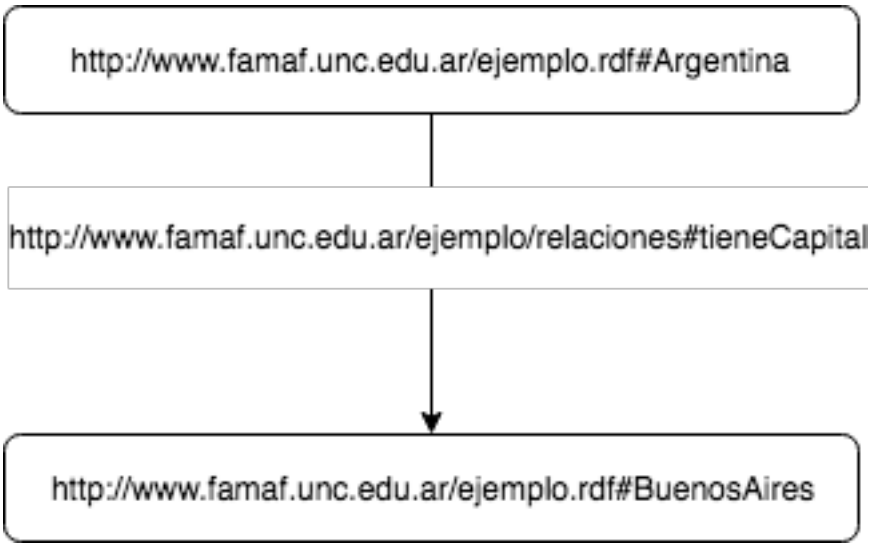


Figure 3.4: Tripla RDF (en representación de grafo) describiendo que Buenos Aires es la capital de Argentina

3.4 OWL

El lenguaje OWL extiende RDF y RDFS. Su objetivo principal es traer el poder expresivo y de razonamiento de las lógicas de descripción a la web semántica. Desafortunadamente, no todo lo expresable en RDF es expresable en $\mathcal{DL}$. Por ejemplo, las clases de clases no están permitidas en todos los tipos de $\mathcal{DL}$, y algunas expresiones de triplas no tendrían sentido en $\mathcal{DL}$. Es por eso que OWL puede solo ser una extensión sintáctica de RDF/RDFS (RDFS es una extensión sintáctica y semántica de RDF).

3.4.1 *Semántica de OWL $\mathcal{DL}$*

Un concepto en $\mathcal{DL}$ es referido como una clase en OWL y un rol en $\mathcal{DL}$ es una propiedad en OWL. Para describir una ontología en OWL, o una $\mathcal{KB}$, puede utilizarse la sintaxis de $\mathcal{DL}$. En los cuadros 3.1, 3.2 y 3.3 veremos cómo escribir expresiones de $\mathcal{DL}$ en OWL.

3.5 DE $\mathcal{DL}$ HACIA $\mathcal{ML}$

Luego de muchas definiciones y desarrollo teórcico en $\mathcal{DL}$ y $\mathcal{ML}$, ya contamos con todos los ingredientes que necesitamos para poder hacer una traducción correcta de fórmulas en $\mathcal{DL}$ a fórmulas en $\mathcal{ML}$

In the eagerness of finding symmetries on Description Logics ($\mathcal{DL}$), a set of tools for finding symmetries in Modal Logics ($\mathcal{ML}$) are to be used. But first, a bridge between the two logis needs to be build: the translations are the approach to build such bridge.

3.5.1 *Función de traducción* Ψ

Trabajaremos con la sintaxis de OWL para definir nuetra función. Sea $\mathcal{R}$ una propiedad y $\mathcal{C}, \mathcal{C}_1$ y $\mathcal{C}_2$ clases, sea $\mathcal{F}_{\mathcal{DL}}$ el conjunto de fórmulas en $\mathcal{DL}$ y $\mathcal{F}_{\mathcal{ML}}$ el conjunto de fórmulas en $\mathcal{ML}$, definamos Ψ recursivamente de la siguiente manera:

$$\Psi : D_\Psi \subseteq \mathcal{F}_{\mathcal{DL}} \rightarrow \mathcal{F}_{\mathcal{ML}}$$

$$\begin{aligned}
\Psi(\mathcal{R}\, someValuesFrom\, \mathcal{C}) &\doteq \langle\mathcal{R}\rangle\Psi(\mathcal{C}) \\
\Psi(\mathcal{R}\, allValuesFrom\, \mathcal{C}) &\doteq [\mathcal{R}]\Psi(\mathcal{C}) \\
\Psi(\mathcal{R}\, hasValue\, \mathcal{C}) &\doteq [\mathcal{R}]\Psi(\mathcal{C}) \wedge \langle\mathcal{R}\rangle\Psi(\mathcal{C}) \\
\Psi(\mathcal{R}^{-} someValuesFrom\, \mathcal{C}) &\doteq \langle\mathcal{R}^{-}\rangle\Psi(\mathcal{C}) \\
\Psi(\mathcal{C}_1 \sqsubseteq \mathcal{C}_2) &\doteq \forall(\Psi(\mathcal{C}_1) \implies \Psi(\mathcal{C}_2)) \\
\Psi(\mathcal{C}_1 \equiv \mathcal{C}_2) &\doteq \Psi(\mathcal{C}_1 \sqsubseteq \mathcal{C}_2) \wedge \Psi(\mathcal{C}_2 \sqsubseteq \mathcal{C}_1) \\
\Psi(\mathcal{C}_1 \cup \mathcal{C}_2 &\doteq \Psi(\mathcal{C}_1) \vee \Psi(\mathcal{C}_2) \\
\Psi(\mathcal{C}_1 \cap \mathcal{C}_2) &\doteq \Psi(\mathcal{C}_1) \wedge \Psi(\mathcal{C}_2) \\
\Psi(\mathcal{R}\, ObjectDomain\, \mathcal{C}) &\doteq \forall(\langle\mathcal{R}\rangle\top \implies \Psi(\mathcal{C})) \\
\Psi(\mathcal{R}\, ObjectRange\, \mathcal{C}) &\doteq \forall(\langle\mathcal{R}^{-}\rangle\top \implies \Psi(\mathcal{C})) \\
\Psi(functional\, \mathcal{R}) &\doteq \forall(\langle\mathcal{R}\rangle\top \implies [\mathcal{R}]\top) \\
\Psi(inverseFunctional\, \mathcal{R}) &\doteq \forall(\langle\mathcal{R}^{-}\rangle\top \implies [\mathcal{R}^{-}]\top) \\
\Psi(\mathcal{C}_1\, Disjoint\, \mathcal{C}_2) &\doteq \forall(\Psi(\mathcal{C}_1) \Leftrightarrow \neg\Psi(\mathcal{C}_2)) \\
\Psi(Complement\, \mathcal{C}) &\doteq \neg\Psi(\mathcal{C}) \\
\Psi(Cardinality\, type\, \mathcal{N}\, \mathcal{C}) &\doteq \langle str(type\mathcal{N}\mathcal{C})\rangle\Psi(\mathcal{C})
\end{aligned}$$

SINTAXIS ABSTRACTA	SINTAXIS EN $\mathcal{DL}$	SEMÁNTICA
	Descripciones (C)	
A(Referencia URI)	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
owl:Thing	$\top$	owl:Thing$^{\mathcal{I}} = \Delta^{\mathcal{I}}$
owl:Nothing	$\bot$	owl:Nothing$^{\mathcal{I}} = \emptyset^{\mathcal{I}}$
intersectionOf($C_1 C_2$)	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
unionOf($C_1 C_2$)	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
complementOf(C)	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
restriction(R hasValue(o))	$R : o$	$\{x \mid (x, o^{\mathcal{I}}) \in R^{\mathcal{I}}\}$
restriction(R allValuesFrom(C))	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y.(x,y) \in R^{\mathcal{I}} \Rightarrow y \in$
restriction(R someValues-From(C))	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y.(x,y) \in R^{\mathcal{I}} \cup y \in$
restriction(R minCardinality(n))	$\geq nR$	$\{a \in \Delta^{\mathcal{I}} \mid \lvert\{b \mid (a,b) \in R^{\mathcal{I}}\}\rvert \geq$
restriction(R maxCardinality(n))	$\leq nR$	$\{a \in \Delta^{\mathcal{I}} \mid \lvert\{b \mid (a,b) \in R^{\mathcal{I}}\}\rvert \leq$
	Rango de Datos (D)	
D (referencia URI)	D	$D^{\mathcal{D}} \subseteq \Delta_{\mathcal{D}}^{\mathcal{I}}$
oneOf($v_1, ...$)	$\{v_1, ..., \}$	$\{v_1^{\mathcal{I}}, ..., \}$
	Propiedades de Objetos (R)	
R (referencia URI)	R	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
	R^-	$(R^{\mathcal{I}})^-$
	Propiedades de Datos (U)	
U (referencia URI)	U	$U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta_{\mathcal{D}}^{\mathcal{I}}$
	Individuos (o)	
o (referencia URI)	o	$o^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
	Valores de Datos (v)	
v (literal RDF)	v	$v^{\mathcal{D}}$

Table 3.1: $\mathcal{DL}$ OWL descripciones, rango de datos, propiedades, individuos y valores de datos con sintaxis y semántica

SINTAXIS ABSTRACTA	SINTAXIS EN $\mathcal{DL}$	SEMÁNTICA
	Clases	
Class(A partial $C_1...C_n$)	$A \sqsubseteq C_1...C_n$	$A^{\mathcal{I}} \subseteq C_1^{\mathcal{I}} \cap ... \cap C_n^{\mathcal{I}}$
Class(A total $C_1...C_n$)	$A \equiv C_1...C_n$	$A^{\mathcal{I}} = C_1^{\mathcal{I}} \cap ... \cap C_n^{\mathcal{I}}$
SubClassOf(C_1C_2)	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
intersectionOf(C_1C_2)	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
EquivalentClasses($C_1...C_n$)	$C_1 \equiv ... \equiv C_n$	$C_1^{\mathcal{I}} = ... = C_n^{\mathcal{I}}$
DisjointClasses($C_1...C_n$)	$C_i \sqcap C_j = \bot, i \neq j$	$C_i^{\mathcal{I}} \sqcap C_j^{\mathcal{I}} = \emptyset, i \neq j$
Datatype(D)		$D \subset \Delta_{\mathcal{D}}^{\mathcal{I}}$
	Anotaciones	
AnnotationProperty(S)		
	Individuos	
Individual(		
o type(C_1)...type(C_n)	$o \in C_i$	$o^{\mathcal{I}} \in C_i^{\mathcal{I}}$
value(R_1o_1)....value(R_no_n)	$\{o, o_i\} \in R_i$	$\{o^{\mathcal{I}}, o_i^{\mathcal{I}}\} \in R_i^{\mathcal{I}}$
value(U_1v_1)...value(U_nv_n))	$\{o, v_i\} \in U_i$	$\{v^{\mathcal{I}}, v_i^{\mathcal{I}}\} \in U_i^{\mathcal{I}}$
SameIndvidual($o_1...o_n$)	$o_1 = ... = o_n$	$o_1^{\mathcal{I}} = ... = o_n^{\mathcal{I}}$
DifferentIndividual($o_1...o_n$)	$o_i \neq o_j, i \neq j$	$o_i^{\mathcal{I}} \neq o_j^{\mathcal{I}}, i \neq j$

Table 3.2: $\mathcal{DL}$ OWL Axiomas y hechos

SINTAXIS ABSTRACTA	SINTAXIS EN $\mathcal{DL}$	SEMÁNTICA
	Propiedades de Tipos de Datos	
DatatypeProperty(		
U super(U_1)...super(U_n)	$U \sqsubseteq U_i$	$U^{\mathcal{I}} \subseteq U_i^{\mathcal{I}}$
domain(C_1)...domain(C_m)	$\geq 1U \sqsubseteq C_i$	$U^{\mathcal{I}} \subseteq C_i^{\mathcal{I}} \times \Delta_{\mathcal{D}}^{\mathcal{I}}$
range(D_1)...range(D_l)	$\top \sqsubseteq \forall U.D_i$	$U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times D_i^{\mathcal{I}}$
[Functional])	$\top \sqsubseteq \leq 1U$	U_i es funcional
SubPropertyOf($U_1 U_2$)	$U_1 \sqsubseteq U_2$	$U_1^{\mathcal{I}} \subseteq U_2^{\mathcal{I}}$
	Propiedades de Objetos	
ObjectProperty(		
Rsuper(R_1)...super(R_n)	$R \sqsubseteq R_i$	$R^{\mathcal{I}} \subseteq R_i^{\mathcal{I}}$
domain(C_1)...domain(C_m)	$\geq 1R \sqsubseteq C_i$	$R^{\mathcal{I}} \subseteq C_i^{\mathcal{I}} \times \Delta_{\mathcal{D}}^{\mathcal{I}}$
range(C_1)...range(C_l)	$\top \sqsubseteq \forall R.C_i$	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times C_i^{\mathcal{I}}$
[inverseOf(R_0)]	$R \equiv (R_0^-)$	$R^{\mathcal{I}} = (R_0^{\mathcal{I}})^-$
[Summetric]	$R \equiv (R^-)$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^-$
[Functional]	$\top \sqsubseteq \leq 1R$	$R^{\mathcal{I}}$ es funcional
[inverseFunctional]	$\top \sqsubseteq \leq 1R^-$	$(R^{\mathcal{I}})^-$ es funcional
[Transitive])	$Tr(R)$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^+$
SubPropertyOf($R_1 R_2$)	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
EquivalentProperties(R_1...R_n)	$R_1 \equiv ... \equiv R_n$	$R_1^{\mathcal{I}} = ... = R_n^{\mathcal{I}}$

Table 3.3: Axiomas y hechos

Parte II

EXPERIMENTACIÓN

4

HERRAMIENTAS [4-5 HOJAS] IMPLEMENTACIÓN

4.1 SCRIPTS

Los principales:

- init.sh:
  - mkdirs.sh: crea los directorios para las salidas de las herramientas sobre lógica modal
  - unzip.sh: descomprime las ontologías para trabajar
  - wipe.sh: borra datos previos (se puede correr init.sh luego de haber obtenido algunos resultados)
  - buid.sh: compila todas las herramientas para lógica modal
- sbt.sh:
  - Se encarga de realizar las traducciones de todas las ontologías que encuentre en el directorio *ontologies/* a lógica modal. También genera un mapeo entre variables proposicionales y los nombres de los conceptos
- doall.sh:
  - k.sh: todas las fórmulas en lógica modal que resolvió nuestro traductor, las pasa a formato CNF
  - s.sh: genera los grafos (no?)
  - b.sh: busca automorfismos en los grafos generados por el script anterior
  - bp.sh: bliss-proc.py
  - map.sh: mapea de vuelta las variables proposicionales a los nombres de los conceptos

Algunos scripts adicionales:

- info.sh:
  - Se encarga de generar archivos de información sobre las ontologías, como tamaño (MB), tbox count, abox count, etc. La información se guarda en <*dir*>
- generate_sheet.py:
  - Genera un archivo con toda la información obtenida por los scripts, de forma estructurada.

4.2 BLISS

Aprender

4.3 MANUAL DE USO

https://github.com/giovannirescia/thesis

4.4 PIPELINE

Nuestro input será una ontología, por lo que primero empezaremos trabajando con lógicas de descripción. El primer paso para poder aplicar todas las herramientas para encontrar simetrías es hacer uso de las traducciones definidas en la sección **??** y luego ir aplicando cada uno de los scripts.

4.4.1 *dl2ml (traducción)*

Este script traduce de DL a LM.

4.4.2 *kcnf*

Este script transforma una fórmula de lógica modal a una versión CNF.

4.4.3 *sy4ncl*

4.4.4 *bliss*

finding automorphism
http://www.tcs.hut.fi/Software/bliss/

4.4.5 *bliss proc*

5

ANÁLISIS DE SIMETRIAS EN ONTOLOGIAS EXISTENTES [5 HOJAS]

5.1 DESCRIPCIÓN DE LAS ONTOLOGÍAS UTILIZADAS

sda Tabla 5.1
hola Tabla 5.2

5.2 TABLA DE DATOS

5.3 ANÁLISIS DE DATOS

5.4 CONCLUSIÓN: ANDA

Ontología	TBOX	Time	Generators	total time
Family	1	0.2	22	2
Galen	2	0.4	33	2

Table 5.1

ONTOLOGÍA	TBOX	GENERATOR
Family	1	3
Galen	2	4
Dolce	3	5

Table 5.2: Resultados

Parte III

CONCLUSIÓN

6 CONCLUSIONES & TRABAJO FUTURO [3 HOJAS]

6.1 RESUMEN DE LO HECHO (DAR LOS LIMITES DEL TRABAJO)

6.2 "OPINIONES"

6.3 QUÉ MÁS SE PUEDE HACER A PARTIR DE ACÁ

7

CONCLUSIONES PERSONALES SOBRE EL TRABAJO (OPTATIVA)

BIBLIOGRAFÍA

[1] Donini et al. *The Complexity of Concept Languages*. 1991.

[2] Lavesque Brachman. «The Tractability of Subsumption in Frame-Based Description Languages». En: (1984).

[3] Copeland. *Computability: Turin, Gödel, Chuch, and Beyond*. The MIT Press, 2015 (Reprint).

[4] Francesco M. Donini. *Foundations of Knowledge Representation and Reasoning*. 1994.

[5] Thomas Gruber. *A Translatin Approach to Portable Ontology Specification*. Inf. téc. Computer Science Department, Stanford University, Stanford, California, 1993.

[6] Thomas A. Lipkis. «A KL-ONE Classifier». En: (1982).

[7] Marvin Minsky. «A Framework for Representing Knowledge». En: (1981).

[8] Ezequiel Orbe. «SLM». Tesis doct. FaMAF, 2014.

[9] Quillian. *Semantic Memory Models*. 1967.

[10] Alfred Tarski. *Introduction to Logic: and to the Methodology of Deductive Sciences*. 1995.