BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA


FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

# "Modelado de problemas de actualización utilizando un enfoque de actualización de secuencias de programas lógicos basado en Answer Set Programming"

TESIS PROFESIONAL

presentada por Mayra Cordero Romero
para obtener el grado de Licenciada en Ciencias de la Computación

ASESORES
Dra. Claudia Zepeda Cortés
Dr. José Luis Carballido Carranza

# Agradecimientos

Le agradezco a Dios por mostrarme día a día que con humildad, paciencia y sabiduría todo es posible.

A mis padres Juan Cordero Quiñones y Adela Romero Galicia por el apoyo incondicional que me brindaron a lo largo de la Licenciatura.

A mi hermano Juan Manuel Cordero Romero por ser un ejemplo de lucha y esfuerzo.

A mi hermana Lorena Cordero Romero por sus motivaciones y buen sentido del humor, por ser mi consejera y amiga.

A mi cuñada Isabel Portada Flores por su aliento para seguir adelante, por su amistad y consejos.

A mi sobrino Josue Cordero Portada por todas esas sonrisas y amor incondicional que me ha brindado.

A José Rodolfo Vázquez Fuentes por motivarme a seguir adelante, por todo su apoyo, compañia y consejos.

A todos mis profesores de la licenciatura que me enseñaron tanto de la profesión como de la vida, impulsandomé siempre a seguir adelante.

A mi asesora de tesis, Dra. Claudia Zepeda Cortés por su asesoría y dirección en el

trabajo de investigación, así como también por su apoyo y amistad que me permitieron aprender muchas más cosas no solo relacionadas con el proyecto.

A la institución CONACYT que me otorgo una beca que me permitió alcanzar este logro.

A mis familiares y amigos por el apoyo moral, su amistad, amor, consejos, animo, y compañia.

A todas esas personas que han confiando en mí y que de alguna manera me han apoyado.

A TODOS USTEDES MI AGRADECIMIENTO Y BENDICIONES

# Resumen

La actualización es un tema que hoy en día se presenta en todos los aspectos del mundo, es un hecho por demás evidente e innegable que el mundo en su conjunto se encuentra sometido a un proceso de cambio continuo y hablar de actualización significa hablar de cambios, hablar de actualidad. Actualizar implica renovar, reemplazar, restablecer, restaurar o modificar alguna actividad, información o tarea retrasada en algo reciente, de manera más general, podemos decir que la actualización consiste en convertir información retrasada en actual, ponerla al día o proporcionar datos distintos a los ya aportados.

El siguiente trabajo de investigación consiste en analizar y modelar un conjunto de problemas que tienen que ver con actualización de la información para lo cuál haremos uso de un enfoque de actualización basado en secuencias de programas lógicos y que utiliza a la semántica estable (también conocida como Answers Set Programming) como base. Los problemas que consideramos estan definidos en diferentes dominios. Para el modelado y el análisis de los problemas implementamos diferentes programas utilizando el software que obtienen modelos estables o aswer sets conocido como DLV.

Finalmente realizamos una discusión y comparación de resultados obtenidos en los diferentes dominios de prueba.

# Índice general

# Índice de figuras

# Capítulo 1

# Introducción

De manera general sabemos que la actualización nos permite modificar o remplazar algo, esto es necesario ya que vivimos en constante cambio donde la actualización es inevitable. Por ejemplo, en una empresa se actualiza información de clientes, estados de cuenta, ventas, pagos, salarios, etc., pero no sólo en este tipo de situaciones se requiere de actualización. El termino actualizar támbien puede aplicarse en otras áreas como por ejemplo la Inteligencia Artificial(IA).

La Inteligencia Artificial(IA) es una de las principales áreas de interés en problemas de carácter no monótono, principalmente en el tema conocido como "paradigma de agentes", este paradigma está interesado en el desarrollo de agentes inteligentes que puedan actuar de forma autónoma y razonada. Los agentes inteligentes, se conforman por una base de conocimientos llamado creencias donde estas creencias representan sus conocimientos sobre el entorno, sobre sí mismo y sobre otros agentes. Como los agentes inteligentes deben responder de forma racional, es posible que su base de conocimientos obtenga nueva información, donde esta información modifique e incluso invalide los conocimientos previos que tenía almacenados el agente inteligente.

Por ejemplo, si la base de conocimientos $P1$ de un agente inteligente contiene reglas $-a$ y $b$, y el agente inteligente recibe nueva información $P2$ que consiste en una regla $a$ entonces, es evidente que la simple unión de $P1$ y $P2$ genera una inconsistencia [6]. Por lo tanto, es necesario aplicar algún método de actualización para actualizar

$P1$ con $P2$ y al mismo tiempo, que el enfoque sea capaz de tener en cuenta que las reglas de $P2$ tienen mayor prioridad que las reglas de $P1$.

El objetivo de esta tesis es estudiar las propiedades y características de un enfoque de actualización basado en en secuencias de programas lógicos y Answer Set Programming, para lo cual modelaremos un conjunto de problemas de diferentes dominios utilizando el enfoque de actualización. Nuestro enfoque para las actualizaciones se basa en Answer Set Programming (ASP) [6]. ASP es una representación del conocimiento declarativo y el lenguaje de programación lógica que nos permite manejar problemas con conocimientos por defecto y permite producir un razonamiento no monótono utilizando el concepto de negación como fracaso. Destacamos el hecho de que la semántica de actualización que utilizamos en esta tesis nos permite expresar actualizaciones en términos de fórmulas en vez de sólo literales, es decir, se trata de un enfoque acerca de las secuencias de actualización de las teorías generales teniendo la capacidad de considerar secuencias de actualización de programas lógicos, $(P1, P2, .., Pn)$, con $n > 2$. Cabe destacar que una de las características importantes de este enfoque de actualización es su propiedad para tratar los cambios con el fin de evitar inconsistencias.

## 1.1. Objetivos

### 1.1.1. Objetivo General

Modelar un conjunto de problemas de actualización para diferentes dominios, utilizando un enfoque de actualización basado en secuencias de programas lógicos y en Answer Set Programming.

### 1.1.2. Objetivos Específicos

1. Estudiar, analizar y comprender un enfoque de actualización de información basado en programación lógica para secuencias de programas lógicos.

2. Analizar y modelar un conjunto de problemas de actualización para diferentes dominios, utilizando el enfoque de actualización basado en secuencias de programas lógicos y Answers Set Programming.

3. Implementar los problemas seleccionados usando DLV para calcular los Answer Sets.

4. Analizar las implementaciones para concluir si el enfoque utilizado debe o no ser extendido. El análisis estará basado en la experiencia obtenida durante el desarrollo del modelado y solución de los problemas seleccionados.

## 1.2. Estructura de la Tesis

**Capítulo 2: Marco Teórico** presentamos de manera formal los conceptos principales para el desarrollo de la investigación. Los temas de estudio son Tipos de Razonamiento no Monotónico, Abducción, Agentes Inteligentes, Características de Agentes Inteligentes, Programación Lógica, Lenguajes Lógicos de Programación, Answer Set Programming y Modelos Estables, Tipos de Negación, Semántica de Actualización, Mínimos Generalizados Answer Sets, Mínimos Extendidos Generalizados Answer Sets y Secuencias de Actualización.

**Capítulo 3: Modelado de Problemas de Actualización** presentamos el desarrollo de los problemas a los cuales aplicamos el enfoque de actualización. Los programas los divididos de la siguiente manera para su mejor entendimiento:

*a*) *Descripción del Problema*
Describimos el problema que se desea resolver por medio del enfoque de actualización. Cabe destacar que cada problema tiene dominio diferente.

*b*) *Modelado del Problema Mediante un Lenguaje Formal*
Analizamos y representamos la narrativa del problema por medio de un lenguaje formal llamado lógica proposicional.

*c*) *Aplicación del Enfoque de Actualización al Problema*
Aplicamos el enfoque de actualización sobre el problema para construir el programa de actualización llamado $P\oslash$.

*d*) *Obtención de los Extedidos Explicitos Generalizados Answer Sets (EEGS)*
Obtenemos el conjunto de extendidos explicitos generalizados answer sets utilizando el resolvedor de answers sets llamado dlv.

*e*) *Obtención de los Mínimos Extedidos Explicitos Generalizados Generalizados Answer Sets (MEEGS)*
Obtenemos los mínimos extendidos explicitos generalizados answer sets, por medio de la cardinalidad y contensión.

*f*) *Discusión de Resultados*
Analizamos los resultados obtenidos y verificamos si son realmente soluciones satisfactorias con respecto a nuestras expectativas.

*g*) *Conclusiones*
Presentamos las conclusiones de cada problema.

**Capítulo 4: Conclusiones y Trabajos a Futuro** presentamos las conclusiones y trabajos a futuro sobre la investigación, así como también tres apéndices, apéndice A, apéndice B y apéndice C donde colocamos todos los extendidos explicitos generalizados answer sets correspondientes a los problemas y finalmente colocamos las referencias.

# Capítulo 2

# Marco Teórico

En este Capítulo presentamos un panorama general de las ideas y conceptos principales tratados en este trabajo de investigación. Incluimos los conceptos principales como Tipos de Razonamiento no Monotónico, Abducción, Agentes Inteligentes, Características de Agentes Inteligentes, Programación Lógica, Lenguajes Lógicos de Programación, Answer Set Programming y Modelos Estables, Tipos de Negación, Semántica de Actualización, Mínimos Generalizados Answer Sets, Mínimos Extendidos Generalizados Answer Sets y Secuencias de Actualización.

## 2.1. Tipos de Razonamiento no Monotónico

El razoamiento no mónotono es muy apropiado cuando se da alguna de las siguientes situaciones:

*a*) el conocimiento que se posee es incompleto

*b*) el universo de discurso es cambiante

*c*) las suposiciones que se formulan son temporales

Las lógicas no monótonas son sistemas en los cuales la adición de nuevos axiomas puede invalidar teoremas concluidos anteriormente. El desarrollo de estas lógicas tiene especial importancia en el campo de la IA, ya que permite estudiar el razonamiento basado en el sentido común, modelar reglas con excepciones, actualizar nuestras creencias después de nuevos descubrimientos, o tener en cuenta las omisiones. En la vida real, la gente cambia fácilmente su concepto sobre las cosas, procediendo a ajustar su creencia (opinión) para evitar errores. De este modo, las viejas conclusiones son abandonadas y la aparición de nuevas evidencias provoca la adición de nuevas creencias. En resumen podemos decir que el razonamiento basado en sentido común es no monótono. El razonamiento por defecto o razonamiento de sentido común, es un tipo de razonamiento que permite establecer conclusiones a partir de información parcial, posteriormente esta información parcial puede desecharse cuando se obtiene nueva información relevante.

Por ejemplo, si de un determinado animal se sabe que es un ave, y no se conoce nada más, se puede asumir que es capaz de volar. No obstante, este hecho debe ser retractado si después se sabe que ese determinado animal es un pingüino. Las lógicas por defecto permiten representar proposiciones como "si x es un ave, entonces x puede volar, a menos que haya algo que lo contradiga". El razonamiento abductivo es el sistema que deriva las explicaciones más probables de los hechos conocidos. En este tipo de razonamiento puede ocurrir que las explicaciones más probables no son necesariamente las correctas. Por ejemplo, la explicación más probable de ver el pasto mojado es que ha llovido; sin embargo, esta explicación debe ser retractada cuando se sepa que la causa real de que el pasto estuviera mojado era un rociador. En tanto que la primera explicación (llovió) es retractada debido a la adición de información (se activó un rociador).

### 2.1.1. Abducción

La abducción es una inferencia y se puede entender de modo operativo de la siguiente forma: la abducción es un proceso por medio del cual se crea, evalúa

y se compureba una hipótesis de carácter ampliativo, con un alto grado de ser pausible y de responder de modo acertado a los hechos o fenómenos que no han podido ser catalogados dentro de un marco teórico. La abducción al tratar de brindar explicaciones pasa de ser una inferencia a constituirse en un proceso lógico no lineal, que permite el ingreso de variables externas al sistema de análisis lógico para encontrar posibles alternativas en la solución de un problema. Plantea soluciones alternas que amplían los marcos de referencia, logrando un mayor espectro interpretativo de los fenómenos. Pierce asume que la abducción es a su vez un proceso epistémico de adqusición de conocimiento, por medio del cual se hace posible el convertir una conjetura en una afrimación; el hacer conjeturas sobre el mundo que nos rodea es escencial para poder sobrevivir y adaptarse a un entorno [4].

## 2.2. Agentes Inteligentes

El nacimiento de la Inteligencia Artificial trajo consigo una gran diversidad de tecnologías, destacando entre ellas la representación del conocimiento, sistemas de razonamiento y sistemas de aprendizaje. A partir de los años 90s y con la implementación de las anteriores tecnologías marcaron el inicio de los Agentes Inteligentes. Actualmente, se puede apreciar cada vez más su uso en distintos campos no sólo en el científico, también con fines comerciales. El desafío, sigue siendo la confiabilidad para delegarles tareas y que la toma de decisiones sea la correcta.

Para ello se requiere sistemas computacionales con razonamiento monotónico y no monotónico. Este último, ha despertado gran interés debido a que se debe buscar el mecanismo óptimo para representar el conocimiento dinámico, siendo la actualización de creencias uno de los principales mecanismos para brindar razonamiento a un agente inteligente en un ambiente de incertidumbre [13].

En el campo de la Inteligencia Artificial (IA), la siguiente definición propuesta por Wooldridge  Jennings en[15] , es de las más aceptadas.

Definición 1. Sistemas computacionales que habitan en algún ambiente dinámi-

co y complejo, pudiendo sentir ese entorno y actuar en consecuencia, tomando en cuenta el conjunto de objetivos y motivaciones que intenta conseguir a través de sus acciones.

Para algunos autores como Cherniak en [1], consideran como punto de partida para describir la característica o propiedad mínima que debe contar un sistema computacional para que sea considerado agente es la racionalidad y, se debe cumplir aún cuando su racionalidad sea mínima [11]. Para que un agente sea racional implica el decidir o elegir, lo que conlleva a los siguientes procesos: deseos y creencias.

### 2.2.1. Características de Agentes Inteligentes

Autores como Franklin Graesser [5] y, Wooldridge Jennings [2] , señalan que las características básicas de un agente serán el ser racional, autónomo, reactivo, pro-activo y social, motivo por el cual sólo se describirán éstas. Cabe destacar que el nivel de abstracción serán las características con que cuente ese agente.

Racional. Es la propiedad que posee un agente inteligente de tomar una decisión y que además sea correcta, con el objetivo de optimizar sus tareas para llegar a su tarea definida. Es decir, realize la tarea asignada y para ello tome la mejor opción con base en su conocimiento.

Autónomo. Es la capacidad que tiene un agente de tomar decisiones propias, sin necesidad de que alguien lo guie. Aún, cuando llegue a tomar una decisión errónea debe ser capaz de buscar una alternativa que lo lleve a su meta.

Reactivo. Se mencionaba que al ser autónomo un agente puede cometer errores, ello le permite alimentar su base de conocimiento y aprender de su experiencia, que en un futuro sus decisiones sean correctas, consistentes con su base de conocimiento y tenga una mejor visión de su medio ambiente.

Pro-activo. Un agente no sólo debe actuar ante eventos producidos por su ambiente, también debe de ser capaz de tomar la iniciativa para satisfacer sus objetivos.

Social. La capacidad de poder relacionarse y comunicarse con otros agentes.

## 2.3. Programación Lógica

El modelo usado tradicionalmente para enfrentar un problema con ayuda de la computadora consiste, primero, en realizar un análisis y estudio del problema para, después, proponer un algoritmo que, ejecutado paso a paso, sea capáz de encontrar las soluciones del problema en cuestión. La computadora, en este caso, es utilizada simplemente como una herramienta de cálculo que realiza, de manera más rápida y eficiente, las operaciones y cómputos requeridos por el algoritmo propuesto. La programación lógica ofrece un modelo alternativo para enfrentar y tratar de solucionar problemas auxiliados por la computadora. Este modelo, lo que pretende, es proveer a la computadora de una descripción del problema que tratamos y no propiamente de un mecanismo para construir soluciones. La computadora será entonces la responsable de analizar todos los casos posibles y determinar cuales de ellos representan soluciones al problema descrito.
El esquema de la programación lógica necesita entonces de un lenguaje que nos permita describir problemas a la computadora. Este lenguaje debe ser claro y específico, falto de ambigüedad. Además debe ser capaz de expresar construcciones matemáticas típicas. El lenguaje que, por supuesto, mejor se adapta a estos requerimientos es el lenguaje de la lógica matemática. Por medio de este lenguaje se escriben los programas lógicos que entendemos como teorías proposicionales. Una teoría proposicional es sólo un conjunto finito de fórmulas bien formadas. Se le puede llamar sólo teoría o programa.

El lenguaje formal que utilizamos para el modelado de los programas en esta investigación pertenece a la lógica proposicional y está formado por: símbolos proposicionales: $m, n, p, q, r, s, t, ...$, conectivos proposicionales: $\wedge, \vee, \leftarrow, \bot, -, \neg$ y símbolos auxiliares: ‘(’,‘)’,‘.’.
También consideramos los dos tipos de negación: negación fuerte o clásica escrita como $(-)$ y negación débil o fracaso escrita como $(\neg)$, en el apartado 2.4.1

hablaremos de la diferencia entre estos dos tipos de negaciones.
Asumimos que para cualquier fórmula proposicional $f$ , $\neg f$ es una abreviatura de $f \leftarrow \bot$. Un átomo es un símbolo proposicional. Un literal es un átomo $a$ o la negación de un átomo $a$. Dado un conjunto de átomos $A$, definimos $-A = \{-l \,|\, l \in A\}, \neg A = \{\neg l \,|\, l \in A\}$, y $Lit_A = A \cup -A$. La signatura de un programa lógico $P$ denotado como $L\rho$, es el conjunto de átomos que se producen en $P$. Por lo general, la sintaxis de fórmulas bien formadas dentro de los programas lógicos se ha limitado a la forma $f \leftarrow g$ que es sólo otra manera de escribir $g \rightarrow f$. Las fórmulas bien formadas $f \leftarrow g$ se llaman reglas donde $f$ se llama la cabeza y $g$ el cuerpo de la regla. Una regla con un cuerpo vacío, $f \leftarrow$, se llama un hecho, y una regla sin cabeza $\bot \leftarrow g$ se llama restricción. Hechos y restricciones también se denominan $f$ y $\leftarrow g$ respectivamente.

Las reglas disyuntivas son de particular interés para nosotros. Una regla disyuntiva es la regla cuya cabeza es una disyunción y tiene la siguiente forma:

$$A_1 \vee ... \vee A_k \leftarrow B_1 \wedge ... \wedge B_m \wedge ... \wedge \neg C_1 \wedge ... \wedge \neg C_n \tag{2.1}$$

Donde $k > 0$, $Ai's$, $Bj's$ y $Ck's$ son átomos. Un programa lógico disyuntivo es sólo un conjunto finito de reglas disyuntivas, se le puede llamar simplemente programa disyuntivo en el que no se presenta ninguna ambigüedad. Siguiendo la notación tradicional de la programación lógica, a veces *not* se utiliza en lugar de $\neg$, y $a, b$ en lugar de $a \wedge b$ . Así, la regla disyuntiva (2.1) puede reescribirse como: $A_1 \vee ... \vee A_k \leftarrow B_1, ..., B_m, not\ C_1\ , ...,\ not\, C_n$ Sin embargo, utilizamos nuestra propia notación. Así que volvemos a escribir la regla disyuntiva (2.1) como:

$$A_1 \vee ... \vee A_k \leftarrow B_1, ..., B_m, \neg C_1, ..., \neg C_n$$

Una regla disyuntiva con $k = 1$ se llama una regla normal. Queremos hacer hincapié en el hecho de que en nuestro enfoque, un programa lógico se interpreta como una teoría proposicional.

### 2.3.1. Lenguajes Lógicos de Programación

Los lenguajes lógicos están dentro de la categoría de la programación declarativa. En ella las instrucciones son basadas en reglas lógicas, las cuales dan una descripción de los objetos de un dominio y sus propiedades. Este conjunto de instrucciones es a menudo llamado base de conocimiento y la meta de la programación declarativa es encontrar modelos o consecuencias de esta base. La base de conocimiento es implementada en lenguajes de programación denominados lógicos, algunos de los más importantes son PROLOG, DATALOG, DLV y SMODELS. Nosotros utilizaremos DLV.

## 2.4. Answer Set Programming y Modelos Estables

Para que un programa lógico tenga un signficado y pueda ser interpretado por herramientas computacionales se le debe asignar una semántica. Una semántica es, intuitivamente, una manera de determinar el tipo de conclusiones que se pueden establecer a partir de un conjunto de reglas. Un problema debe codificarse entonces, en la forma de un programa lógico, buscando que la semántica del programa capture, precisamente, las soluciones del problema en cuestión.

La semántica de modelos estables (stable models), introducida por Gelfond y Lifschitz en[7], significó un importante adelanto en el área de la programación lógica. Una de las principales novedades de la semántica de modelos estables fue la facilidad con que permitía modelar el concepto de negación como falla en los programas lógicos. Este tipo de negación facilita la especificación de conocimiento por omisión (default knowledge) y establecer relaciones inerciales que son fundamentales para modelar sistemas dinámicos. Los modelos estables, como fueron definidos en[8], proveen de un significado semántico a los programas lógicos que, sin embargo, están condicionados a una sintaxis particular muy restringida. La semántica de modelos estables consideraba, únicamente, reglas

de la forma:

$$h_1 \vee ... \vee h_n \leftarrow b_1 \wedge ... \wedge b_m \wedge \neg b_1 \wedge ... \wedge \neg b_{m+1}$$

donde cada $h_i$ y $b_i$ es una fórmula atómica con $n \geq 0, m \geq 0$ y $l \geq 0$. Este tipo de reglas, aunque representan sólo una clase restringida de programas lógicos, probaron ser de gran utilidad para poder enfrentar y resolver una amplia cantidad de problemas. La semántica de answer sets, propuesta por Lifschitz, Tang y Turner en[14], generaliza la noción de modelos estables para una clase de programas más amplia. Los programas lógicos que se consideran en esta semántica permiten reglas de la forma $H \leftarrow B$, donde el par de fórmulas $H$ y $B$ pueden estar compuestas de conjunciones,disyunciones y negaciones arbitrariamente anidadas. En[3], se muestran evidencias claras de cómo este lenguaje puede ser de gran utilidad para representar y resolver problemas reales. Así surgió la Programación con Answer Sets (Answer Set Programming, ASP) o también conocida como A-Prolog. Este paradigma representa la acumulación de una importante cantidad de avances en razonamiento no-monótono, inteligencia artificial y programación lógica durante los últimos 15 años. El éxito de la semántica de los modelos estables, y posteriormente de los answer sets, se debió en gran medida a su capacidad para resolver problemas. Gracias a la existencia de implementaciones eficientes para calcular modelos estables como DLV y SMODELS la gama de problemas que podían enfrentarse con este nuevo paradigma creció rápidamente para incluir problemas combinatorios, establecer y resolver problemas de planeación, modelar el comportamiento de agentes lógicos y aplicaciones de inteligencia artificial en general. Sin embargo ocurre que, de un modo ciertamente sorprendente, las primeras relaciones entre la lógica proposicional y la programación lógica estaban simplemente restringidas al uso de un lenguaje común. Las definiciones de las primeras semánticas, aunque ciertamente están basadas en nociones de la lógica matemática, están más bien propuestas a partir de métodos que simplemente parecen razonables para asignar modelos a un programa lógico dado. El estudio de propiedades o caracterizaciones de la semántica en términos de lógicas proposicionales es un tema de investigación mucho más reciente.

### 2.4.1. Tipos de Negación

Una de las características innovadoras que presenta la semántica de answer sets es, precisamente, el uso de dos tipos de negaciones distintas. La negación clásica o explícita, que denotaremos con el conectivo $-$, permite especificar explícitamente conocimiento que sabemos de antemano es falso. Mientras que la negación como falla, denotada como $\neg$, tiene un significado un tanto distinto. La intuición de la negación como falla es que una fórmula $\neg F$ es cierta si, con la información disponible, no es posible mostrar que la fórmula $F$ sea cierta. Observe que el no poder mostrar que un hecho es cierto no implica necesariamente que éste sea falso. Puede ocurrir que, por ejemplo, el hecho sea en efecto cierto pero la información disponible no sea suficiente para demostrarlo. Entre las ventajas que ofrece la negación como falla está, como lo habíamos mencionado antes, la facilidad con que se puede expresar conocimiento por omisión. Ejemplo 1.1. Una de las aplicaciones donde se utilizan ambos tipos de negaciones es para describir situaciones que normalmente son ciertas, pero que pueden tener excepciones. Podemos escribir, por ejemplo, que *"normalmente las aves vuelan"* utilizando la regla: $vuela(X) \leftarrow ave(X) \wedge \neg - vuela(X)$. De manera un poco más formal esta regla dice: *"Si $X$ es un ave y, con la información disponible, no es posible determinar que no vuela entonces asumiremos que sí puede volar"* . Supongamos que tenemos la siguiente información acerca de algunas aves en nuestra base de conocimientos:
$ave(pato)$.
$ave(pelícano)$.
$ave(pingüino)$.
$-vuela(pingüino)$.
El sistema podrá inferir correctamente que tanto el pato como el pelícano, como son aves, pueden volar. Sin embargo el sistema no tratará de usar el mismo razonamiento para demostrar que los pingüinos vuelan pues ya que sabe, de manera explícita, que no lo hacen. Es importante hacer la aclaración de que, en diversas fuentes importantes en la literatura, es común encontrar al símbolo *not* para denotar a la negación como falla (aquí denotada como $\neg$) y al símbolo $-$ para la negación explícita (aquí denotada $-$). La notación que aquí empleamos

sigue de una corriente iniciada por trabajos recientes que buscan establecer relaciones entre las lógicas proposicionales y la propia programación lógica. En este caso resulta que la negación $\neg$ utilizada comúnmente en la lógica proposicional coincide precisamente con la negación como falla de la programación lógica. El uso de la negación explícita $-$ en la semántica de answer sets es, de hecho, muy restringido. Sólo se permiten ocurrencias de esta negación dentro de fórmulas atómicas. Además de que no juega un papel realmente significativo en la definición de la semántica.

## 2.5. Semántica de Actualización

En esta sección presentamos la semántica de actualización, el tema principal de está tesis [10].

### 2.5.1. Mínimos Generalizados Answer Sets

En esta subsección presentamos algunas definiciones básicas de sintaxis y semántica para programas lógicos abductivos. Los programas lógicos abductivos son programas que nos permiten representar razonamiento abductivo, es un tipo de razonamiento que a partir de la descripción de un hecho o fenómeno ofrece o llega a una hipótesis, la cual explica las posibles razones o motivos del hecho mediante las premisas obtenidas. Charles Sanders Peirce la llama una conjetura. Esa conjetura busca ser, a primera vista, la mejor explicación, o la más probable. En la abducción a fin de entender un fenómeno se introduce una regla que opera en forma de hipótesis para considerar dentro de tal regla al posible resultado como un caso particular. En otros términos: en el caso de una deducción se obtiene una Conclusión $q$ de una premisa $p$, mientras que el razonar abductivo consiste en explicar $q$ mediante $p$ considerando a $p$ como hipótesis explicativa.

**Definición 1.1.** Un programa lógico abductivo es un par $\langle P, A \rangle$ donde $P$ es un programa arbitrario y $A$ un conjunto de átomos, llamados abductivos.

La noción de mínimo generalizado answer set es usada para definir la semántica de programas lógicos abductivos.
**Definición 1.2.** $\langle M(\triangle)\rangle$ es un explicito generalizado answer set del programa abductivo $\langle P, A\rangle$ si solo si $\triangle \subseteq A$ y $M(\triangle)$ es un answer set de $P \cup \triangle$.

**Definición 1.3.** Definimos un orden entre los explicitos generalizados answer sets de un programa abductivo de la siguiente manera: Sean $M(\triangle 1)$ y $M(\triangle 2)$ answer sets generalizados de $\langle P, A\rangle$, definimos $M(\triangle 1) \leq M(\triangle 2)$ si $\triangle 1 \subseteq \triangle 2$.

**Definición 1.4.** $M(\triangle)$ es un mínimo explicito generalizado answer set de $\langle P, A\rangle$ si solo si $M(\triangle)$ es un explicito generalizado answer set de $\langle P, A\rangle$ y es un mínimo abductivo con respecto al orden de inclusión.

Por efectos prácticos dado un mínimo explicito generalizado answer set $M(\triangle)$ sólo estamos interesados en la primera entrada es decir en $M$ y lo llamamos mínimo generalizado answer set de un programa lógico abductivo.

### 2.5.2. Mínimos Extendidos Generalizados Answer Sets

La semántica para actualizaciones consiste en una secuencia de programas basados en una extensión de la definición de mínimos extendidos generalizados answer sets. La extensión se llama mínimos extendidos generalizados (MEG) answer sets y se describe en esta sección. Se introduce la definición de programas extendidos abductivos lógicos (EAL) . Está definición es similar a la definición 1.1, pero se agrega una función sobreyectiva. Desde un programa EAL obtenemos sus extendidos explicitos generalizados (EEG) answer sets. Utilizamos la función sobreyectiva de un programa de EAL para definir un orden entre sus EEG answer sets y obtener sus MEG answer sets.

**Definición 1.5.** Un programa extendido abductivo lógico (EAL) es un triple $\langle P, A, f\rangle$ tal que $P$ es un programa arbitrario, $A$ es un conjunto de átomos, y $f$ es alguna función sobreyectiva con dominio $A$ y codominio $\{1, ...N\}, N > 0$

Ejemplo 1: Sea $\langle P, A, f \rangle$ un programa (EAL) donde $A$ es el conjunto de átomos $\{x_1^1, x_2^1, x_1^2, x_2^2, \}$ $f : A \to \{1, 2\}$ es la función $f(x_j^i = i)$, y $P$ el siguiente programa:

```
a  ← not x11.
b  ← a, not x12.
-a ← not x21.
-b ← not x22.
c  ←.
```

Aquí la definición de extendido explicito generalizado answer set.

**Definición 1.6.** Sea $\langle P, A, f \rangle$ un programa (EAL), $M$ un conjunto de literales y $\triangle \subseteq A$. Un extendido explicito generalizado (EEG) answer set de $\langle P, A, f \rangle$ es un par $\langle M, \triangle \rangle$ si $M$ es un answer set de $P \cup \triangle$.

Ejemplo 2: Vamos a considerar el programa EAL $\langle P, A, f \rangle$ del Ejemplo 1. El cuadro 2.1 muestra los diferentes EEG answer sets de $\langle P, A, f \rangle$ con sus respectivos $\triangle \subseteq A$.

| $\triangle$ | $\langle M, \triangle \rangle$ |
|---|---|
| {} | no tiene EEG answer sets |
| {x11} | $\langle\{x11, -a, -b, c\}, \{x11\}\rangle$ |
| {x12} | no tiene EEG answer sets |
| {x21} | no tiene EEG answer sets |
| {x22} | no tiene EEG answer sets |
| {x11, x12} | $\langle\{x11, x12, -a, -b, c\}, \{x11, x12\}\rangle$ |
| {x11, x21} | $\langle\{x11, x21, -b, c\}, \{x11, x21\}\rangle$ |
| {x11, x22} | $\langle\{x11, x22, -a, c\}, \{x11, x22\}\rangle$ |
| {x12, x21} | $\langle\{x12, x21, a, -b, c\}, \{x12, x21\}\rangle$ |
| {x12, x22} | no tiene EEG answer sets |
| {x21, x22} | $\langle\{x21, x22, a, b, c\}, \{x21, x22\}\rangle$ |
| {x11, x12, x21} | $\langle\{x11, x12, x21, -b, c\}, \{x11, x12, x21\}\rangle$ |
| {x11, x12, x22} | $\langle\{x11, x12, x22, -a, c\}, \{x11, x12, x22\}\rangle$ |
| {x11, x21, x22} | $\langle\{x11, x21, x22, c\}, \{x11, x21, x22\}\rangle$ |
| {x12, x21, x22} | $\langle\{x12, x21, x22a, c\}, \{x12, x21, x22\}\rangle$ |
| {x11, x12, x21, x22} | $\langle\{x11, x12, x21, x22, c\}, \{x11, x12, x21, x22\}\rangle$ |

Cuadro 2.1: EEG answer sets del programa $\langle P, A, f\rangle$ del ejemplo 1.

Ahora presentamos el orden entre los EEG answers sets de un programa de EAL. Antes de presentar la definición formal de este orden damos la intuición detrás de él. Primero tenemos que recordar que un programa de EAL es una triple, $\langle P, A, f\rangle$, basado en un programa lógico $P$, un conjunto de átomos $A$, y una función sobreyectiva con dominio $A$ y codominio $\{1, ..., N\}$. Esta función se utiliza para definir una partición del conjunto $A$ en subconjuntos de $N$, es decir, $\{a \in A | f(a) = i\} para 1 \leq i \leq N$. Una de las partes de $A$, es decir $A_k$, y el conjunto de criterio de inclusión son usados para obtener el más pequeño de cualquiera de dos EEG answer sets, $\langle M, \triangle_1\rangle$ y $\langle M, \triangle_2\rangle$ de un programa EAL $\langle P, A, f\rangle$. Así que si $\langle M, \triangle_1\rangle$ es menor que $\langle M, \triangle_2\rangle$ entonces: $(\triangle_1 \cap A_k) \subset (\triangle_2, \cap A_k)$, y para todo $j, k < j \leq N$, $(\triangle_1 \cap A_j) = (\triangle_2 \cap A_j)$.
La siguiente definición indica formalmente este orden de inclusión.

| $\triangle$ | $\langle M, \triangle \rangle$ |
|---|---|
| $\langle \emptyset, \emptyset \rangle$ | no tiene EEG answer sets |
| $\langle \emptyset, \{x11\} \rangle$ | $\langle \{x11\}, \{-a, -b, c\} \rangle$ |
| $\langle \emptyset, \{x12\} \rangle$ | no tiene EEG answer sets |
| $\langle \{x21\}, \emptyset \rangle$ | no tiene EEG answer sets |
| $\langle \{x22\}, \emptyset \rangle$ | no tiene EEG answer sets |
| $\langle \emptyset, \{x11, x12\} \rangle$ | $\langle \{x11, x12, -a, -b, c\}, \{x11, x12\} \rangle$ |
| $\langle \{x21\}, \{x11\} \rangle$ | $\langle \{x11, x21, -b, c\}, \{x11, x21\} \rangle$ |
| $\langle \{x22\}, \{x11\} \rangle$ | $\langle \{x11, x22, -a, c\}, \{x11, x22\} \rangle$ |
| $\langle \{x21\}, \{x12\} \rangle$ | $\langle \{x12, x21, a, -b, c\}, \{x12, x21\} \rangle$ |
| $\langle \{x22\}, \{x12\} \rangle$ | no tiene EEG answer sets |
| $\langle \{x21, x22\}, \emptyset \rangle$ | $\langle \{x21, x22, a, b, c\}, \{x21, x22\} \rangle$ |
| $\langle \{x21\}, \{x11, x12\} \rangle$ | $\langle \{x11, x12, x21, -b, c\}, \{x11, x12, x21\} \rangle$ |
| $\langle \{x22\}, \{x11, x12\} \rangle$ | $\langle \{x11, x12, x22, -a, c\}, \{x11, x12, x22\} \rangle$ |
| $\langle \{x21, x22\}, \{x11\} \rangle$ | $\langle \{x11, x21, x22, c\}, \{x11, x21, x22\} \rangle$ |
| $\langle \{x21, x22\}, \{x12\} \rangle$ | $\langle \{x12, x21, x22, a, c\}, \{x12, x21, x22\} \rangle$ |
| $\langle \{x21, x22\}, \{x11, x12\} \rangle$ | $\langle \{x11, x12, x21, x22, c\}, \{x11, x12, x21, x22\} \rangle$ |

Cuadro 2.2: EEG answer sets del programa $\langle P, A, f \rangle$ del ejemplo 1.

**Definición 1.7.** Sea $\langle P, A, f \rangle$ un programa EAL donde el codominio de $f$ es el conjunto $\{1, ..., N\}$, $N > 0$. Sea $A_i = \{a \in A | f(a) = i\}$. Establecemos un orden de inclusión entre los EEG answer sets de $\langle P, A, f \rangle$ de la siguiente manera: Sea $\langle M_1, \triangle_1 \rangle$ y $\langle M_2, \triangle_2 \rangle$ dos EGG answer sets de $\langle P, A, f \rangle$, definimos $\langle M_1, \triangle_1 \rangle \leq_{i\ nclu} \langle M_2, \triangle_2 \rangle$ si y sólo si existe $k$, $1 \leq k \leq N$ tal que $(\triangle_1 \cap A_k) \subset (\triangle_2, \cap A_k)$, y para toda $j, k < j \leq N$, $(\triangle_1 \cap A_j) = (\triangle_2 \cap A_j)$.
Ejemplo 3: Vamos a considerar el programa EAL $\langle P, A, f \rangle$ del Ejemplo 1 y sus EEG answer sets del cuadro 2.1. Vemos que $N = 2$ , $A_1 = \{x_1^1, x_2^1\}$ y $A_2 = \{x_1^2, x_2^2\}$ con $f(x_j^i = i)$. Podemos comprobar que
$\langle \{x_2^1, x_1^2, x_2^2, a, c\}, \{x_2^1, x_1^2, x_2^2\} \rangle \leq_{i\ nclu} \langle \{x_1^1, x_2^1, x_1^2, x_2^2, c\}, \{x_1^1, x_2^1, x_1^2, x_2^2\} \rangle$; ya que $k = 1$ tal que $(\{x_2^1, x_1^2, x_2^2\} \cap A_1) \subset (\{\mathrm{x}_1^1, x_2^1, x_1^2, x_2^2\} \cap A_1)$, es decir $\{x_2^1\} \subset \{\mathrm{x}_1^1, x_2^1\}$, y para todo $j$, $1 < j \leq 2$ tenemos que $(\{x_2^1, x_1^2, x_2^2\} \cap A_2) = (\{x_1^1, x_2^1, x_1^2, x_2^2\} \cap A_2)$, es decir $\{x_1^2, x_2^2\} = \{x_1^2, x_2^2\}$; o
$\langle \{x_1^1, x_2^1, -a, -b, c\}, \{x_1^1, x_2^1\} \rangle \leq_{i\ nclu} \langle \{x_1^1, x_2^1, x_1^2, -b, c\}, \{x_1^1, x_2^1, x_1^2\} \rangle$ ya que $k =$

2 tal que $\{x_1^1, x_2^1\} \cap A_2 \subset \{x_1^1, x_2^1, x_1^2\} \cap A_2$, es decir $\emptyset \subset \{x_2^1\}$, y no hay $j, 2 < j \leq 2$.

Alternativamente, y con el fin de hacer más fácil de entender cómo funciona el orden entre los EEG answer sets de un programa de EAL podríamos reescribir el cuadro 2.1 como el cuadro 2.2. En el cuadro 2.2 cada subconjunto $\triangle$ del cuadro 2.1 se reescribe como un par donde sus entradas corresponden a los subconjuntos de $\triangle$ en orden descendiente lexicográfico con respecto al superíndice de cada átomo en $\triangle$. Por ejemplo $\{x_2^2, x_1^1, x_2^1\}$ se reescribe como $\langle\{x_2^2\}, \{x_1^1, x_2^1\}\ \rangle$. En el cuadro 2.1 es más facil comparar los pares de EEG answer sets, ya que dado dos EEG answer sets sólo verificamos si el subconjunto de la izquierda es subconjunto de otro, en otro caso se compara si el subconjunto de la izquierda son los mismos EEG's en este caso sólo verificamos si el segundo subconjunto con respecto a un orden lexicográfico descendiente es subconjunto del otro, y así sucesivamente. Así que de acuerdo al cuadro 2.1, podemos verificar que $\langle\{x12, 21, 22, a, c\}, \{x12, x21, x22\}\rangle \leq_{i\ nclu} \langle\{x11, x12, x21, x22, c\}, \{x11, x12, x21, x22\}\rangle$ ya que corresponden a los pares $\langle\{x21, 22, \}, \{x12\}\rangle$ y $\langle x21, 22, \}, \{x11, x12\}\rangle$donde el primer subconjunto es el mismo y es segundo subconjunto es tal que $\{x12\} \subset \{x11, x12\}$.

También es posible definir un orden de cardinalidad entre los EEG answer sets de un programa $\langle P, A, f\rangle$, denotado por $\leq_{c\ ard}$. Esta definición puede obtenerse a partir de definición 1.7 reemplazando conjunto criterio de inclusión por el criterio conjunto de cardinalidad.

**Definición 1.8.** Sea $\langle P, A, f\rangle$ un programa EAL donde el codominio de $f$ es el conjunto $\{1, ..., N\}$, $N > 0$. Sea $A_i = \{a \in A | f(a) = i\}$. Establecemos un orden de cardinalidad entre los EEG answer sets de $\langle P, A, f\rangle$ de la siguiente manera: Sea $\langle M_1, \triangle_1\rangle$ y $\langle M_2, \triangle_2\rangle$ dos EGG answer sets de $\langle P, A, f\rangle$, definimos $\langle M_1, \triangle_1\rangle \leq_{c\ ard} \langle M_2, \triangle_2\rangle$ si hay un $k$, $1 \leq k \leq N$ tal que $|\triangle_1 \cap A_k| \subset |\triangle_2, \cap A_k|$, y para toda $j, k < j \leq N$, $|\triangle_1 \cap A_j| = |\triangle_2 \cap A_j|$.
Veremos cómo utilizar cardinalidad y establecer órdenes de inclusión para obtener los mínimos EEG answer sets de un programa dado EAL.
**Definición 1.9.** Sea $\langle P, A, f\rangle$ un programa EAL $\langle M, \triangle\rangle$ es un $k$ mínimo extendido explicito generalizado ($MEEG_k$) answer set de $\langle P, A, f\rangle$ ( donde $k$ $\in \{$inclu, card$\}$) $si \langle M, \triangle\rangle$ es un EEG answer set de $\langle P, A, f\rangle$ y no hay otro

EEG anwer set $\langle M', \triangle' \rangle$ de $\langle P, A, f \rangle$ tal que $\langle M', \triangle' \rangle \leq_k \langle M, \triangle \rangle$. A efectos prácticos, dado un s MEEG$_k answerset$,$\langle M, \triangle \rangle$ estamos interesados en su primera entrada, es decir $M$, y lo llamamos k-mínimo extendido generalizado ($MEG_k$) answer set de un programa EAL. Ejemplo 4: Vamos a considerar el programa $\langle P, A, f \rangle$ del ejemplo 1. Según el cuadro 2.1, se puede verificar que $\langle \{x11, -a, -b, c\}, \{x11\} \rangle$ es el único $MEEG_{inclu}$ answer set de $\langle P, A, f \rangle$, y el $MEG_{inclu}$ answer set es $\{x11, -a, -b, c\}$.
Observación: En el resto de esta tesis sólo se utilizará el orden de inclusión para los EEG answer sets. Por lo tanto, vamos a escribir $\leq$ para denotar este orden y $MEEG$ o $MEG$ para denotar un $MEEG_{inclu}$ o un $MEG_{inclu}$ answer set. Hay algunas propiedades simples de los MEG answer sets que serán de utilidad para mostrar algunas de las propiedades de actualización. Dado un programa de EAL $\langle P, A, f \rangle$, las propiedades de los MEG answer set indican lo siguiente: -en caso de que el programa lógico $P$ tenga al menos un answer set entonces,

- Si $\langle M, \triangle \rangle$ es un MEEG answer set de un programa EAL entonces el subconjunto de abducibles, $\triangle$, es el conjunto vacío.
- Si $M$ es un MEG answer set de un programa EAL entonces $M$ no incluye átomos abducibles.

- en caso de que el programa lógico de $P$ tenga al menos un answer set entonces, $M$ es un answer set de $P$ si y sólo si $M$ es un MEG answer set del programa de EAL. -en caso de que $f$ sea una función sobreyectiva con el dominio $A$ y codominio $\{1\}$ los mínimos generalizados answer set de un programa lógico Abductivo, coinciden con los MEEG answer sets de un programa EAL como se define en este proyecto.

### 2.5.3. Secuencias de Actualización

En esta subsección hablaremos del operador de actualización detotado por $\oslash$ capaz de tratar con las secuencias de actualización. Formalmente , se dice que

una secuencia de actualización a través de $L\rho$ denotada como $\mathbf{P}$, es una secuencia de $n$ programas lógicos, $(P1, ..., Pn)$, donde $L\rho$, representa el conjunto de átomos que aparecen en $\bigcup( 1 \leq i \leq n)\ P_i$ y $N_P i$ es el número de reglas del programa lógico de la secuencia $P_i$ sin tener en cuenta las restricciones. La semántica de nuestro operador de actualización se basa en los mínimos extendidos generalizados answer sets de un programa extendido abductivo lógico en particular, $\langle P_{\oslash}, B, f\rangle$ donde $B$ y $f$ son un conjunto particular de átomos y una función sobreyectiva particular, respectivamente, $P_{\oslash}$ se define en términos de los de n-programas lógicos dados, es decir $P_{\oslash} = P_1 \oslash ... \oslash P_n$. Ahora definimos este programa como extendido abductivo lógico.

**Definición 1.10.** Sea $\mathbf{P} = (P_1, ...P_n)$ una secuencia de actualización a través de $L\rho$. Sea $n = \sum_{i=1}^{n-1} N_{pi}$ y $B$ un conjunto de $n$ nuevos átomos abducibles tal que $B = \{b_j^i, 1 \leq i \leq n-1, 1 \leq j \leq N_{pi}$, y $b_j^i \notin L\rho\}$. Sea $L^*\rho = L\rho \cup B$. Definimos el programa lógico de actualización $P_{\oslash}$ a través de $L^*\rho$ que consiste en lo siguiente:

*a*) Todas las restricciones en $P_1, ..., P_{n-1}$.

*b*) Para cada $r_j \in P_i, 1 \leq i \leq n-1, 1 \leq j \leq N_{pi}$ añadimos la regla $r_j \leftarrow \neg b_j^i$, donde $b_j^i \in B$.

*c*) Todas las reglas y restricciones $r \in P_n$.

Definimos el programa EAL de $P$ como triple $\langle P_{\oslash}, B, f\rangle$ donde $B$ es el conjunto de nuevos átomos abducibles de $P_{\oslash}$ y $f\colon B \to \{1, ..., n-1\}$ es la función sobreyectiva donde $f(b_j^i = i)$.

La semántica de nuestro operador de actualización se basa en los MEG answer sets de un programa EAL de una secuencia de actualización determinada.

**Definición 1.11.** Sea $\langle P_{\oslash}, B, f\rangle$ el programa EAL de una secuencia de actualización $\mathbf{P}$ sobre $L\rho$. $M$ es un $\oslash$-actualización answer set de $\mathbf{P}$, si $M'$ es un MEG answer set de $\langle P_{\oslash}, B, f\rangle$ y $M = M' \cap L\rho$.

Ejemplo 5: Sea $\mathbf{P}$=$(P1, P2, P3)$ una secuencia de actualización a través de $L\rho$= $\{a, b, c\}$ donde $P1$,$P2$ y $P3$ son los siguientes programas:

$P1$:

```
a    ← .
b    ← a.
```
$P2$:
```
-a   ← .
-b   ← a.
```
$P3$:
```
c    ← .
```
Sea $B$ el conjunto e nuevos átomos abucibles $\{x_1^1, x_2^1, x_1^2, x_2^2\}$. Así el programa lógico de actualización $P_{\oslash} = P1 \oslash P2 \oslash P3$ sobre $L^*\rho$ es el siguiente programa lógico:
```
a    ← not x_1^1.
b    ← a, not x_2^1.
-a   ← not x_1^2.
-b   ← not x_2^2.
c    ← .
```
El programa EAL de $\mathbf{P}$ es el triple $\langle P_{\oslash}, B, f \rangle$, donde $f\colon B \to \{1, 2\}$ es la función $f(b_j^i = i)$. Podemos comprobar que el programa EAL de $\mathbf{P}$ corresponde al programa EAL del Ejemplo 1. Por otra parte, por ejemplo 4 sabemos que su único MEG answer set es $\{x_1^1, -a, -b, c\}$. Por lo tanto, $\{-a, -b, c\}$ es el único $\oslash$-actualización answer set de $\mathbf{P}$. El siguiente ejemplo ilustra támbien la definición de $\oslash$-actualización answer sets. Este ejemplo corresponde al Ejemplo 1 de [9].

Ejemplo 6: Consideremos la secuencia de actualización $P = (P1, P2)$ donde $P1$:
```
sleep   ← night, not watchTv, not other.
night   ← .
tvOn    ← tvBroke.
watchTv ← tvOn.
```
$P2$:
```
-tvOn         ← powerFailure.
-tvOn         ← assignmentDue, working.
assignmentDue ← .
```

`working` $\leftarrow$ `.`

`other` $\leftarrow$ `working.`

El programa EAL de $\mathbf{P}$ es el triple $\langle P_{\oslash}, B, f \rangle$, donde $f\colon B \to \{1\}$ es la función $f(b_j^i = i)$. Podemos comprobar que el único MEG answer set de $\langle P_{\oslash}, B, f \rangle$ es $\{x_3^1, night, other, assignmentDue, working, -tvOn\}$. Por lo tanto, el único $\oslash$-actualización answer set de $\mathbf{P}$ es $\{night, other, assignmentDue, working, -tvOn\}$.

# Capítulo 3

# Modelado de Problemas de Actualización

En este capítulo presentamos el conjunto de problemas a los cuales aplicamos la semántica de actualización. Estos problemas son de diferentes dominios, el objetivo de estos problemas es ver el comportamiento de las secuencias de actualización de programas lógicos. Los problemas que presentamos se encuentran con la misma estructura para su mejor entendimiento.

## 3.1. Problema 1: Control del Suministro de Agua en un Taller

### 3.1.1. Descripción del Problema

Un operario está encargado de controlar que no se vacíe ninguno de los dos tanques de agua ubicados en la parte superior de un taller. Estos dos tanques suministran agua para un subproceso. Cuando alguno de los dos tanques se vacié el operario debe encender la bomba que permite llenar los tanques con el agua almacenada en un tanque cisterna ubicada en el sótano. En caso de que la cisterna se encuentre vacía, la bomba no debe encenderse pues podría dañarse.

Mediante llaves de paso, el operario direcciona el agua, que viene de la bomba, a uno u otro tanque para simplificar el problema, se considera que cada uno de los tanques tiene controles mecánicos el llenado de los tanques. Con la intención de evitar trasladarse periódicamente, para controlar los niveles de agua de los tanques, se decide diseñar un sistema que indique, mediante 3 lámparas, el estado de cada uno de los tanques como se muestra en la figura 3.1. Una de las lámparas debe permanecer encendida cuando el tanque cisterna del sótano se mantenga con agua (lámpara encendida = tanque con agua). Cada una de las otras dos lámparas se deberá encender cuando el tanque correspondiente se vacie (lámpara encendida = tanque sin agua). Para controlar los niveles de agua el sistema debe decidir activar la bomba cuando la lámpara del tanque cisterna esté encendida y, además, se encienda alguna de las otras dos lámparas considerando cuando la cisterna se encuentre vacía.

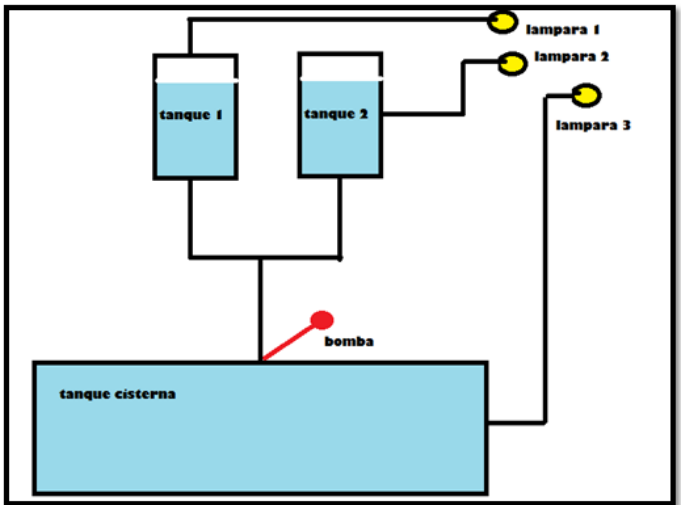


Figura 3.1: Sistema de agua

Como el consumo de agua no es constante, el sistema debe estar preparado para las diferentes situaciones que se le puedan presentar, por ejemplo un situación podría ser que el tanque 1 este vacío y la cisterna con agua, en este caso, el sistema deberá encender la lámpara 1 y al estar encendida la lámpara 1 el sistema deberá activar la bomba para poder llenar nuevamente el tanque 1. Como el sistema debe trabajar con información dínamica el problema surge

cuando el sistema percibe nueva información y está información se contradice con la información anterior, en este caso el sistema debe actualizarse y ser capaz de responder satisfactoriamente.

Un ejemplo de está situación donde se presenta contradicción en la información es el siguiente, que sucedería si inicialmente el sistema indica que el tanque 1 está vacío y al mismo tiempo se presenta nueva información que indica al sistema que la cisterna está vacía.

Si inicialmente el tanque 1 esta vacío entonces el sistema encenderá la lámpara 1 y al encender la lámpara 1 el sistema encenderá la bomba, pero si en ese momento el sistema recibe nueva información que le indica al sistema que la cisterna está vacía entonces el sistema debe decidir no encender la bomba porque puede dañarse, está nueva información que se le presenta al sistema es una contradicción porque al estar vacío el tanque 1 debe encenderse la bomba y al estar vacía la cisterna no debe encenderse la bomba, por lo tanto el sistema debe actualizarse tomando en cuenta la información mas relevante y poder dar una respuesta a esta situación, está actualización del sistema la modelaremos mediante el enfoque de actualización de secuencias de programas lógicos basado en Answer Sets.

### 3.1.2. Modelado del Problema Mediante un Lenguaje Formal

Para llevar acabo el modelado del sistema, representamos la información mediante un lenguaje formal haciendo uso de reglas, hechos y restricciones que mediante programas lógicos nos permiten darle un significado a nuestro problema. El sistema lo representamos mediante 4 programas $P1$, $P2$, $P3$ y $P4$. Explicaremos a grandes rasgos que representa cada programa. El programa $P1$ representa el estado inicial de la lampará 3 que corresponde al estado de la cisterna, el programa $P2$ representa el estado de las lámparas dependiendo de la cantidad de agua de los tanques y la cantidad de agua de la cisterna, el programa $P3$ indica el estado de la bomba dependiendo de las lámparas y

la cisterna y el programa $P4$ indica cuando deben ser llenados los tanques y ádemas representa los diferentes escenarios que se pueden presentar al sistema. A continuación explicaremos cada programa con más detalle.

El programa $P1$ contiene una regla, estas regla indica que la lámpara 3 está encendida por lo tanto la cisterna tiene suficiente agua.

$P1$ :

`r1: lampara3_encendida.`

La regla r1 indica que la lámpara 3 está encendida.

El programa $P2$ contiene tres reglas que le permiten al sistema saber el estado de las lámparas dependiendo de la cantidad de agua en los tanques y la cantidad de agua la cisterna.

$P2$ :

`r1: lampara1_encendida` $\leftarrow$ `tanque1_sin_agua.`

`r2: lampara2_encendida` $\leftarrow$ `tanque2_sin_agua.`

`r3: lampara3_encendida` $\leftarrow$ `not cisterna_vacia.`

La regla r1 indica que si el tanque 1 no tiene agua entonces la lámpara 1 debe estar encendida.

La regla r2 indica que si el tanque 2 no tiene agua entonces la lámpara 2 debe estar encendida.

La regla r3 indica que si no hay evidencia que la cisterna este vacía entonces la lámpara 3 debe estar encendida.

El programa $P3$ contiene dos reglas que le permiten saber al sistema el funcionamiento de la bomba:

$P3$ :

`r1: activar_bomba` $\leftarrow$ `lampara1_encendida,lampara3_encendida.`

`r2: activar_bomba` $\leftarrow$ `lampara2_encendida,lampara3_encendida.`

La regla r1 indica que si la lámpara 1 y la lámpara 3 están encendidas entonces el sistema debe activar la bomba.

La regla r2 indica que si la lámpara 2 y la lámpara 3 están encendidas entonces el sistema debe activar la bomba.

El programa $P4$ contiene seis reglas que permiten saber al sistema cuando se lleva acabo el llenado de los tanques y támbien permite colocar la información

reciente que le llega al sistema para llevar acabo su funcionamiento:

$P4$ :

```
r1: llenar_tanque1 ← activar_bomba,lampara1_encendida.
r2: llenar_tanque2 ← activar_bomba,lampara2_encendida.
r3: tanque1_sin_agua.
r4: -activar_bomba     ← cisterna_vacia.
r5: -lampara3_encendida ← cisterna_vacia.
r6: cisterna_vacia.
```

La regla r1 indica que si la bomba esta activada y la lámpara 1 está encendida entonces se debe llenar el tanque 1.

La regla r2 indica que si la bomba esta activada y la lámpara 2 está encendida entonces se debe llenar el tanque 2.

La regla r3 indica que el tanque 1 está sin agua.

La regla r4 indica que si la cisterna está vacía entonces la bomba debe estar desactivada.

La regla r5 indica que si la cisterna está vacía entonces la lámpara 3 debe estar apagada.

La regla r6 indica que la cisterna está vacía.

Haciendo la uníon de estos 4 programas y corriendo el programa resultante nos damos cuenta que dlv no encuentra ninguna solución a nuestro problema como se muestra en la figura 3.2, esto sucede porque cuando el sistema recibió nueva información esta información resulta contradictoria con la información anterior, por lo tanto debemos actualizar el funcionamiento del sistema.

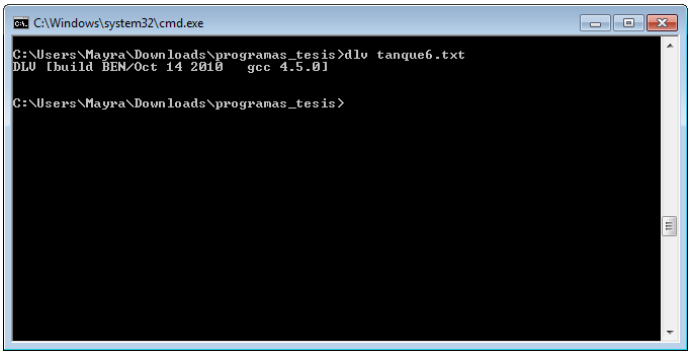


Figura 3.2: Resultado de la unión de programas P1,P2,P3,P4

### 3.1.3. Aplicación del Enfoque de Actualización

El enfoque de actualización se va a encargar de actualizar la información del sistema, tomando en cuenta que la información mas reciente es la información más importante, en este caso es, la información que corresponde al programa $P4$.

Demanera formal y siguiendo la sémantica del enfoque de actualización decimos:
Sea $\mathbf{P}$=$P1, P2, P3, P4$ una secuencia de programas, donde $P1, P2, P3, P4$ son los siguientes programas correspondientes a el funcionamiento del sistema:
$P1$ :

```
r1: lampara3_encendida.
```

$P2$ :

```
r1: lampara1_encendida  ← tanque_1_sin_agua.
r2: lampara2_encendida  ← tanque_2_sin_agua.
r3: lampara3_encendida  ← not cisterna_vacia.
```

$P3$ :

```
r1: activar_bomba  ← lampara1_encendida, lampara3_encendida.
r2: activar_bomba  ← lampara2_encendida, lampara3_encendida.
```

$P4$ :

```
r1: llenar_tanque1  ← activar_bomba, lampara1_encendida.
r2: llenar_tanque2  ← activar_bomba,lampara2_encendida.
r3: tanque1_sin_agua.
r4: -activar_bomba      ← cisterna_vacia.
r5: -lampara3_encendida ← cisterna_vacia.
r6: cisterna_vacia.
```

Y sea $B$ el conjunto de nuevos átomos abductivos $\{x_1^1, x_1^2, x_2^2, x_3^2, x_1^3, x_2^3\}$, el nuevo programa lógico de actualización $P_{\oslash} = P1 \oslash P2 \oslash P3 \oslash P4$ sobre $L^*\rho$ es el siguiente programa:

```
r1: lampara3_encendida ← not x11.
r2: lampara1_encendida ← tanque1_sin_agua, not x21.
```

```
r3: lampara2_encendida ← tanque2_sin_agua, not x22.
r4: lampara3_encendida ← not cisterna_vacia, not x23.
r5: activar_bomba      ← lampara1_encendida, lampara3_encendida,
not x31.
r6: activar_bomba      ← lampara2_encendida, lampara3_encendida,
not x32.
r7: llenar_tanque1     ← activar_bomba,lampara_encendida1.
r8: llenar_tanque2     ← activar_bomba,lampara2_encendida.
r9: tanque1_sin_agua.
r10: -activar_bomba      ← cisterna_vacia.
r11: -lampara3_encendida ← cisterna_vacia.
r12: cisterna_vacia.
```

Una vez obtenido el programa de actualización llevaremos acabo la obtención del conjunto de los EEG answer sets para verificar como fue el comportamiento del sistema.

### 3.1.4. Obtención de los Extendidos Explicitos Generalizados Answer Sets (EEGS)

Ya obtenido el programa actualizado $P_{\oslash}$ y siguendo la sémantica de actualización que indica obtener el conjunto de EEG answer sets por medio de los abductivos utilizando DLV obtenemos el conjunto de EEGS answer sets, este conjunto corresponde al conjunto potencia de los abductivos utilizados para actualizar nuestros programas, este conjunto los presentamos en el apendice A 4.

### 3.1.5. Obtención de los Mínimos Extendididos Generalizados Answer Sets (MEGS)

Una vez teniendo todo el conjunto de EEG answer sets obtenemos por medio de la contensión y la cardinalidad como se muestra en definición 1.9 el MEG answer set correspondiente a el programa actualizado.

Ahora al nuevo programa se le denomina un programa extenido abductivo lógico EAL de **P**, este programa es un triple $\langle P_{\oslash}, B, f\rangle$ donde $f :\rightarrow \{1, 2, 3\}$ es la función $f(b^i_j = i)$. Esta función corresponde al número de abductivos que se utilizaron para actualizar el nuevo programa. Finalmente el MEG answer sets del nuevo programa $\langle P_{\oslash}, B, f\rangle$ es:
$\{x11, tanque1\_sin\_agua, cisterna\_vacia, lampara1\_encendida, -activar\_bomba,$
$- lampara3\_encendida\}$. Nos interesa obtener el MEG answer set ya que buscamos la actualización del sistema omitiendo la más mínima información y como ya mencionamos sólo estamos interesados en $M$, al MEG answer set le quitamos los abductivos que contenga en este caso el abductivo es $x11$ y por lo tanto, el answer set $\oslash$-actualización de **P** es:
$\{tanque1\_sin\_agua, cisterna\_vacia, lampara1\_encendida, -activar\_bomba,$
$- lampara3\_encendida\}$.

### 3.1.6. Discusión de Resultados

Observando el MEG answer set
$\{tanque1\_sin\_agua, cisterna\_vacia, lampara1\_encendida, -activar\_bomba,$
$- lampara3\_encendida\}$ nos podemos dar cuenta que el sistema logró actualizar la información, es por eso que el sistema teniendo en cuenta la información más reciente fue capaz de desactivar la bomba ya que la nueva información que recibió el sistema era que la cisterna se encontraba vacía. Observamos que inicialmente el sistema indico que el tanque 1 estaba vacío y por lo tanto la cisterna deberia activarse para llenar nuevamente el tanque pero al recibir neva información que le indicaba al sistema que la cisterna estaba vacía era necesario actualizar el sistema para evitar esta contradicción y dar una respuesta tomando en cuenta la información más reciente. Al actualizar la información el sistema logro indicar desactivar la bomba ya que aunque el tanque 1 estaba vacío, la información más reciente era que la cisterna se encontraba vacia y por lo tanto se debe desactivar la bomba.

### 3.1.7. Conclusiones

En base a este ejeplo concluimos que la actualización no solo es importante para este sistema, tambieñ podría emplearse en el desarrollo de otros sistemas en los cuales sea necesaria una actualización de información. Teniendo la ventaja que este enfoque permite la manipulación de información dínamica que es una característica de cualquier sistema

## 3.2. Problema 2: Iluminación Inteligente de Oficinas

### 3.2.1. Descripción del Problema

Alrededor del 40 % de toda la electricidad utilizada en las oficinas corresponde a la iluminación. El problema más común que se presenta en las oficinas es el tiempo que permanecen encendidas las luces sin ningún beneficio.
Para solucionar este problema proponemos una iluminación inteligente y eficiente que ayude al ahorro de energía y al cuidado del medio ambiente ya que una buena iluminación genera un adecuado confort visual y ayuda a los usuarios a aumentar la productividad de su trabajo. Proponemos llevar acabo una iluminación inteligente dentro de las oficinas, donde la iluminación se administre por medio de un agente inteligente basado en conocimientos. Este agente inteligente se conformará por sensores, actuadores y lámparas LED para el control de la iluminación.
Un agente inteligente es capaz de realizar tareas por sí mismo reaccionando a su ambiente. Para que los agentes inteligentes lleven a cabo su funcionamiento actúan por medio de sensores, controladores y actuadores. Los sensores son los dispositivos que captan información dependiendo de los parámetros que controlan y una vez captada la información se trasmite la información a un controlador el cual se encarga de procesar la información, actualizarla y generar las ordenes que debe realizar el actuador, en realidad el actuador es la fuerza que impulsa a los dispositivos de los agentes inteligentes a realizar una

acción. Por lo tanto el agente inteligente será el encargado de crear lo ambientes de iluminación necesarios según la situación que se le presente.
El agente inteligente que vamos a modelar es un agente inteligente basado en su conocimiento, por lo que su base de conocimientos puede cambiar, ya que con el agente inteligente debe responder a la situación que se le presente para realizar un funcionamiento correcto.

### 3.2.2. Solución Propuesta

La iluminación una oficina la conforman 4 lámparas que son controladas con un solo switch. Cuando se presiona el switch se encienden las 4 lámparas, al permanecer encendidas las 4 lámparas se produce un exceso de energía insuficiente, por lo tanto proponemos realizar la iluminación por medio de un agente inteligente que controle la iluminación utilizando sensores de presencia. Estos sensores de presencia se van a encargar de ubicar la posición más proxima en la que se encuentra el usuario, una vez que los sensores detecten la presencia y la ubicación del usuario, se encenderá la lámpara más próxima y las otras lámparas permanecerán apagadas para permitir al usuario realizar sus actividades con una buena iluminación y ahorrando energía.
Otro problema que se presenta en las oficinas, es cuando el usuario necesita salir de la oficina por pequeños lapsos de tiempo, por ejemplo, puede salir al baño, a traer un archivo, a una junta, a comer,a sacar copias, etc. y a veces se le olvida apagar la luz, supongamos que si en un edificio existen 30 oficinas y de esas 30 oficinas se les olvida apagar la luz en 10 oficinas, nos podemos dar cuenta que es un exceso de energía insuficiente. Este problema tmabién lo resuelve nuestro agente inteligente ya que si ningún sensor detecta presencia la iluminación se apagará automáticamente. De esta manera controlaremos que solo este prendida la iluminación de la oficina cuando haya un usuario trabajando dentro de la oficina permitiéndole realizar sus actividades diarias de una manera confortable y ahorrando energía.

### 3.2.3. Diseño del Agente Inteligente

El agente inteligente estará conformado por 4 lámparas LED, 4 sensores de presencia y 4 actuadores. Las 4 lámparas LED serán controladas por los 4 sensores de presencia. A los sensores de presencia los representaremos con las siguientes variables $sensor1, sensor2, sensor3, sensor4$ y a las lámparas LED las representaremos como $lampara1$, $lampara2$, $lampara3$ y $lampara4$.
Para el funcionamento del agente inteligente hicimos uso de la lógica proposicional para modelar la base de conocimiento de nuestro agente. La base de conocimiento de nuestro agente se conforma por reglas, hechos y restricciones que nos permiten darle un significado a las acciones que debe realizar el agente.
A continuación explicaremos el funcionamiento correspondiente a los sensores.
El $sensor1$ detectara la presencia del usuario cuando el usuario se encuentre dentro de la oficina en la parte de enfrente de lado derecho, el $sensor2$ detectara la presencia del usuario cuando el usuario se encuentre dentro de la oficina en la parte de enfrente del lado izquierdo, el $sensor3$ detectara la presencia del usuario cuando el usuario se encuentre dentro de la oficina en la parte de atrás de lado derecho y finalmente el $sensor4$ detectara la presencia del usuario cuando el usuario se encuentre dentro de la oficina en la parte de atrás de lado izquierdo. Esto le ayudara al agente inteligente a tener el control y poder ubicar al usuario que dependiendo cuál sea el sensor que detecte presencia serán encendidas las lámparas.
Se encenderán las luces de la siguiente manera:
Si el $sensor1$ detecta presencia se encenderá la $lampara1$.
Si el $sensor2$ detecta presencia se encenderá la $lampara2$.
Si el $sensor3$ detecta presencia se encenderá la $lampara3$.
Si el $sensor4$ detecta presencia se encenderá la $lampara4$.
Como las actividades que que realiza el usuario en la oficina no son estáticas, el agente inteligente necesita actualizar su información para poder generar ambientes confortables que requiera el usuario y para ello, utilizamos el enfoque de actualización de secuencias basado en Aswer Set Programming.

### 3.2.4. Modelado del Problema

Para poder modelar nuestro agente inteligente tenemos que representar la información por medio de programas lógicos. Estos programas lógicos utilizan una gramática formal que nos permite interactuar con la computadora. Los programas lógicos se conformar por reglas, hechos y restricciones que nos van a permitir darle un significado a nuestro problema.
Utilizamos 3 programas lógicos, los cuales representamos con: $P1$, $P2$ y $P3$.
El programa $P1$ lo utilizamos para representar la base de conocimiento inicial de nuestro agente inteligente, esta base de conocimientos corresponde al estado inicial de las lámparas. Para ver el comportamiento del agente suponemos que inicialmente la $lampara1$, $lampara2$, $lampara3$ y $lampara4$ están apagadas. El programa $P2$ lo utilizamos para representar el estado de las lámparas según el detector de presencia, si no tenemos evidencia de que el $sensor1$ detecte presencia entonces asumiremos que la $lampara1$ debe estar apagada, si no tenemos evidencia de que el $sensor2$ detecte presencia entonces asumiremos que la $lampara2$ debe estar apagada, si no tenemos evidencia de que el $sensor3$ detecte presencia entonces asumiremos que la $lampara3$ debe estar apagada y finalmente si no tenemos evidencia de que el $sensor4$ detecte presencia entonces asumiremos que la $lampara4$ debe estar apagada. El programa $P3$ lo utilizamos para representar la información que indica cuando las luces se encenderán dependiendo de donde ubiquen los sensores al usuario, si el $sensor1$ detecta presencia significa que el usuario se encuentra en la parte de enfrente del lado derecho, si el $sensor2$ detecta presencia significa que el usuario se encuentra en la parte de enfrente del lado izquierdo, si el $sensor3$ detecta presencia significa que el usuario se encuentra en la parte de atrás del lado derecho y si el $sensor4$ detecta presencia significa que el usuario se encuentra en la parte de atras del lado izquierdo. Una vez que establecimos al agente en donde podría ubicarse el usuario debemos representar la información que indique como debe ser el encendido de las lamparas, si el usuario se encuentra enfrente de lado derecho entonces el agente inteligente encenderá la $lampara1$, si el usuario se encuentra enfrente de lado izquierdo entonces el agente inteligente encenderá la $lampara2$, si el usuario se encuentra atras de lado derecho entonces el agente

inteligente encenderá la *lampara*3 y si el usuario se encuentra atrás de lado izquierdo entonces el agente inteligente encenderá la *lampara*4. Finalmente colocaremos la base de datos extensional que sirve para representar las diferentes situaciones que se le pueden presentar al agente inteligente, en este ejemplo supondremos que el *sensor*3 detecto presencia.

Ahora representamos la información por medio de programas lógicos de la siguiente manera:

$P1$ :

```
r1: -enc_lamp1.
r2: -enc_lamp2.
r3: -enc_lamp3.
r4: -enc_lamp4.
```

La regla r1 indica que la *lampara*1 está apagada.

La regla r2 indica que la *lampara*2 está apagada.

La regla r3 indica que la *lampara*3 está apagada.

La regla r4 indica que la *lampara*4 está apagada.

$P2$ :

```
r1: -enc_lamp1 ← not sensor1_detecta_presencia.
r2: -enc_lamp2 ← not sensor2_detecta_presencia.
r3: -enc_lamp3 ← not sensor3_detecta_presencia.
r4: -enc_lamp4 ← not sensor4_detecta_presencia.
```

La regla r1 indica que si no hay evidencia que el *sensor*1 detecte presencia entonces el agente inteligente asumirá que la *lampara*1 debe estar apagada.

La regla r2 indica que si no hay evidencia que el *sensor*2 detecte presencia entonces el agente inteligente asumirá que la *lampara*2 debe estar apagada.

La regla r3 indica que si no hay evidencia que el *sensor*3 detecte presencia entonces el agente inteligente asumirá que la *lampara*3 debe estar apagada.

La regla r4 indica que si no hay evidencia que el *sensor*4 detecte presencia entonces el agente inteligente asumirá que la *lampara*4 debe estar apagada.

$P3$ :

```
r1: usuario_enfrenteD ← sensor1_detecta_presencia.
r2: enc_lamp1         ← usuario_enfrenteD.
```

```
r3: usuario_enfrenteI ← sensor2_detecta_presencia.
r4: enc_lamp2         ← usuario_enfrenteI.
r5: usuario_atrasD    ← sensor3_detecta_presencia.
r6: enc_lamp3         ← usuario_atrasD.
r7: usuario_atrasI    ← sensor4_detecta_presencia.
r8: enc_lamp4         ← usuario_atrasI.
r9: sensor3_detecta_presencia.
```

La regla r1 indica que si el $sensor1$ detecta presencia entonces el usuario se encuentra enfrente de lado derecho.
La regla r2 indica que si el usuario se encuentra enfrente de lado derecho entonces debe encenderse la $lampara1$.
La regla r3 indica que si el $sensor2$ detecta presencia entonces el usuario se encuentra enfrente de lado izquierdo.
La regla r4 indica que si el usuario se encuentra enfrente de lado izquierdo entonces debe encenderse la $lampara2$.
La regla r5 indica que si el $sensor3$ detecta presencia entonces el usuario se encuentra atrás de lado derecho.
La regla r6 indica que si el usuario se encuentra atrs de lado derecho entonces debe encenderse la $lampara3$.
La regla r7 indica que si el $sensor4$ detecta presencia entonces el usuario se encuentra atrás del lado izquierdo.
La regla r8 indica que si el usuario se encuentra atrás de lado izquierdo entonces debe encenderse la $lampara4$.
La regla r9 indica que el $sensor3$ detecto presencia.
Ahora haremos uso del enfoque de actualización para actualizar la información y poder ver como es el funcionamiento del agente inteligente.

### 3.2.5. Aplicación del Enfoque de Actualización

De manera formal decimos:
Sea $\mathbf{P}$=$P1, P2, P3$ una secuencia de programas, donde $P1, P2, P3$ son los si-

guientes programas:

$P1$ :

```
r1: -enc_lamp1.
r2: -enc_lamp2.
r3: -enc_lamp3.
r4: -enc_lamp4.
```

$P2$ :

```
r1: -enc_lamp1 ← not sensor1_detecta_presencia.
r2: -enc_lamp2 ← not sensor2_detecta_presencia.
r3: -enc_lamp3 ← not sensor3_detecta_presencia.
r4: -enc_lamp4 ← not sensor4_detecta_presencia.
```

$P3$ :

```
r1: usuario_enfrenteD ← sensor1_detecta_presencia.
r2: enc_lamp1         ← usuario_enfrenteD.
r3: usuario_enfrenteI ← sensor2_detecta_presencia.
r4: enc_lamp2         ← usuario_enfrenteI.
r5: usuario_atrasD    ← sensor3_detecta_presencia.
r6: enc_lamp3         ← usuario_atrasD.
r7: usuario_atrasI    ← sensor4_detecta_presencia.
r8: enc_lamp4         ← usuario_atrasI.
r9: sensor3_detecta_presencia.
```

Y sea $B$ el conjunto de nuevos átomos abductivos $\{x_1^1, x_2^1, x_3^1, x_4^1, x_1^2, x_2^2, x_3^2, x_4^2\}$. El nuevo programa lógico de actualización $P\oslash = P1 \oslash P2 \oslash P3$ el siguiente programa:

```
r1: -enc_lamp1        ← not x11.
r2: -enc_lamp2        ← not x12.
r3: -enc_lamp3        ← not x13.
r4: -enc_lamp4        ← not x14.
r5: -enc_lamp1        ← not sensor1_detecta_presencia, not x21.
r6: -enc_lamp2        ← not sensor2_detecta_presencia, not x22.
r7: -enc_lamp3        ← not sensor3_detecta_presencia, not x23.
```

```
r8: -enc_lamp4        ← not sensor4_detecta_presencia, not x24
r9: usuario_enfrenteD ← sensor1_detecta_presencia.
r10: enc_lamp1         ← usuario_enfrenteD.
r11: usuario_enfrenteI ← sensor2_detecta_presencia.
r12: enc_lamp2         ← usuario_enfrenteI.
r13: usuario_atrasD    ← sensor3_detecta_presencia.
r14: enc_lamp3         ← usuario_atrasD.
r15: usuario_atrasI    ← sensor4_detecta_presencia.
r16: enc_lamp4         ← usuario_atrasI.
r17: sensor3_detecta_presencia.
```

Una vez obtenido el nuevo programa actualizado, debemos obtener el conjunto de todos los EEG answer sets por medio del conjunto potencia de los abductivos y DLV.

### 3.2.6. Obtención de los Extendidos Explicitos Generalizados Answer Sets (EEGS)

Una vez obtenido el nuevo programa $P_{\oslash}$, por medio del conjunto potencia de los abductivos y DLV obtenemos todos los EEG answer sets mostrados en el apendice B 4.

### 3.2.7. Obtención de los Mínimos Extendididos Generalizados Answer Sets (MEGS)

De todo el conjunto de EEG answer sets obtenemos el MEG answer set por medio de la inclusion y cardinalidad como se muestra en definición 1.9. Por lo tanto el nuevo programa EAL de $\mathbf{P}$ es un triple $\langle P_{\oslash}, B, f\rangle$ donde $f :\rightarrow \{1, 2\}$es la función $f(b^i_j = i)$. El MEG answer set de $\langle P_{\oslash}, B, f\rangle$ es:
$\{x13, sensor3\_detecta\_precencia, -en\_lamp1, -en\_lamp2, -en\_lamp4,$
$usuario\_enfrenteD, en\_lamp3\}$ y como solo estamos interesados en $M$ restamos el abductivo que nos permitió la actualización y por lo tanto, el answer

set $\oslash$-actualización es:
$\{sensor3\_detecta\_precencia, -en\_lamp1, -en\_lamp2, -en\_lamp4,$
$usuario\_enfrenteD, en\_lamp3\}$.

### 3.2.8. Discusión de Resultados

Observando el MEG answer set $\{sensor3\_detecta\_precencia, -en\_lamp1,$
$-en\_lamp2, -en\_lamp4, usuario\_enfrenteD, en\_lamp3\}$ nos damos cuenta que el agente inteligente al detectar por medio del sensor 3 presencia de un usuario, el agente inteligente enciende la lampara 3, con este ejemplo vemos como el sistema actualiza su base de conocimientos inicial por medio de la nueva información percibida por el agente inteligente. Al detectar presencia de un usuario por medio del sensor 3 el agente inteligente actualiza el estado de las lamparas que inicialmente se encontraban apagadas y enciende la lampara que en este caso corresponde a la lampara 3. Esto nos permite ver como el sistema se actualiza para responder al escenario que se le presente.

### 3.2.9. Conclusiones

Concluimos que gracias a la actualización los agentes inteligentes puede actualizar su base de conocimiento y así generar respuestas satisfactorias esperadas por los usuarios. Por lo tanto este enfoque es de gran importancia no sólo para el agente de este ejemplo, sino támbien podría ser de gran importancia para agentes inteligentes que uilicen información dínamica o que en algún momento su información sea inconsistente.

## 3.3. Problema 2: Diagnóstico de Enfermedades

### 3.3.1. Descripción del Problema

La medicina ha experimentado avances espectaculares en las últimas décadas. Sin embargo, muchas de las tecnologías y procedimientos que emplea dependen, en última instancia, del factor humano, este es el caso de una fase del protocolo médico habitual llamado diagnóstico.
El proceso diagnóstico tiene, en esencia, un doble objetivo: determinar cuál es la enfermedad del paciente a partir del conjunto de síntomas que éste presenta y descartar posibles diagnósticos alternativos. Es un proceso ciertamente complejo que en gran medida se asemeja a una labor detectivesca. En el caso de los doctores menos experimentados, suelen utilizar principalmente lo que se conoce como sistema 2 de pensamiento. Esta forma de procesar la información es reflexiva, analítica, racional, y supone la aplicación del método hipotético-deductivo a cada caso concreto, a partir de los indicios que el médico observa, genera hipótesis sobre los posibles diagnósticos alternativos que podrían encajar con los síntomas y, a partir de ahí, mediante pruebas, tests y análisis de diversos tipos, va descartando la presencia de enfermedades y encaminándose a "descubrir" qué hay tras las dolencias del paciente. Es un procedimiento altamente sistemático, pero que tiene como contrapartida un alto consumo de recursos cognitivos, además de necesitar más tiempo. Los doctores con una amplia experiencia parecen emplear más habitualmente, sin embargo, el sistema 1 de procesamiento cognitivo, este sistema es intuitivo, automático, y menos demandante en términos cognitivos, además de más rápido. Básicamente, este sistema recurre al uso de "atajos cognitivos", y trata de ver en qué medida los síntomas del paciente se ajustan a los patrones, guiones y prototipos sobre enfermedades que el doctor tiene almacenados en su memoria. Obviamente, esta forma de proceder requiere haber atesorado previamente una amplia experiencia sobre casos y enfermedades, que dotaría al doctor -como se dice habitualmente- con un buen "ojo clínico".

Por fortuna, los doctores emplean ambos sistemas de manera combinada, lo que hace más fiable y seguro todo el proceso de toma de decisiones sobre el diagnóstico. Si tras los primeros intentos de buscar la correspondencia entre síntomas y patrones de enfermedad guardados en la memoria (sistema 1) no se llega a una solución satisfactoria, el doctor recurre entonces a un procedimiento más controlado, el sistema 2; lo que a su vez, a medida que se va ganando en experiencia, irá aumentando el número de patrones y prototipos guardados en la memoria del sistema 1. Parece entonces que, en el caso de los doctores con mayor experiencia, este almacén de memoria contiene más información, y por tanto, su grado de dependencia del sistema 2 es menor, lo que desde un punto de vista cognitivo es muy eficiente.

Tratar de representar este tipo de razonamiento por medio de una computadora no es un proceso ciertamente fácil, este tipo de razonamiento es posible representarlo por medio del razonamiento abductivo. El razonamiento abductivo es un tipo de razonamiento que surge por medio de la lógica no monotónica, este razonamiento es un tipo de razonamiento que a partir de la descripción de un hecho o fenómeno ofrece o llega a una hipótesis, la cual explica las posibles razones o motivos del hecho mediante las premisas obtenidas, es por eso que este tipo de razonamiento es siempre inseguro, pues se basa en una conjetura o sospecha, según Peirce indica en[12]. Pierce establece que un buen diagnóstico significa observar los síntomas del paciente y determinar la enfermedad que más probablemente tenga dicho paciente, de manera formal diríamos *"el conjunto de síntomas C es observado en el paciente; pero si el paciente tuviera la enfermedad A, los síntomas C serían esperables; por tanto, hay razones para sospechar que el paciente tiene la enfermedad A"*. La conclusión abductiva –la enfermedad A, por ejemplo– podria invalidarse si se descubren nuevos hechos –que, por ejemplo, pueden llevarnos a concluir que el paciente no tiene la enfermedad A, sino otra más rara, B, con el mismo conjunto de síntomas C–. Podemos observar que al descubrirse nuevos hechos(síntomas) se debe actualizar la información, para poder dar un nuevo diagnóstico. Si la información no se actualizará podria existir inconsistencia en la información e incluso podria presentarse un diagnóstico incorrecto. Por lo tanto en este ejemplo llevaremos

acabo como se lleva acabo el el diagnóstico de la gripe utilizando el enfoque de actualización basado en Asnswer Set Programming.
La gripe es una infección contagiosa de la nariz, la garganta y los pulmones causada por el virus de la gripe, esta enfermedad comienza de manera abrupta, entre los síntomas frecuentes que se pueden presentar en un paciente se encuentran dolores en el organismo y falta de energía. Algunos pacientes experimentan síntomas tales como mareos o vómitos. El estado febril en los pacientes es usualmente rígido 1 o 2 días, pero puede durar hasta 5 días. En algún punto entre el segundo y el cuarto día de la enfermedad, los síntomas de "todo el organismo" comienzan a desaparecer y los síntomas respiratorios comienzan a incrementarse.

El virus de la gripe se puede establecer en cualquier parte de las vías respiratorias, produciendo síntomas de resfriado, dolor y molestia de garganta, bronquiolitis, infección en el oído o neumonía.

El más sobresaliente de los síntomas respiratorios usualmente es una tos seca. La gran parte de los pacientes también padecen molestias de garganta y cefalea, se conoce como cefalea a los dolores y molestias localizadas en cualquier parte de la cabeza. Son frecuentes la secreción nasal (rinorrea) y estornudos. Estos síntomas (a excepción de la tos) usualmente desaparecen en un ciclo de 4 a 7 días. Suele suceder que algunas veces, el estado febril regresa y la tos y el cansancio usualmente duran semanas después de terminado el resto de la enfermedad.

Algunas veces, suelen confundirse los resfriados con las gripes, ya que los dos comparten algunos síntomas y se presentan en la misma época del año. A pesar de esto, las dos enfermedades son muy distintas. La gran parte de los individuos sufre resfriados varias veces al año y gripe sólo una vez en varios años.
Como ya mencionamos uno de los síntomas que produce la gripe son síntomas correspondientes al resfriado. Este caso, es el problema que planteamos para

este ejemplo. Lo que pretendemos resolver en este problema es ver como se actualiza la información para poder dar un buen diagnóstico si inicialmente el paciente presenta síntomas correspondientes al resfriado y después de un lapso de tiempo presenta síntomas adicionales correspondientes a la gripe. Para un doctor poder diferenciar entre estas dos enfermedades y poder dar un diagnóstico es por medio de su experiencia adquirida y su forma de razonamiento como vimos en un principio, pero para una computadora diferenciar entre estas dos enfermedades y poder dar un diagnóstico no es una tarea sencilla. Lo que buscamos en este caso es poder dar un diagnóstico correcto utilizando el enfoque de actualización de secuencias de programas lógicos basados en Answet Set Programming. Mediante los programas lógicos vamos a representar los síntomas iniciales del paciente así como también los síntomas adicionales o más recientes de dicho paciente y las causas por las cuales un doctor diagnostica un resfriado o gripe. Una de las ventajas de este enfoque de actualizacíon es que nos permite tomar en cuenta la información más reciente, que en este caso serán los síntomas adicionales y en un diagnóstico la información más relevante es la qie se debe tomar en cuenta.

Para modelar nuestro ejemplo, tomamos como referencia la información publicada por el Colegio Oficial de Farmacéuticos de Valencia quienes publicaron la información del cuadro 3.1, donde cada fila de este cuadro indica, para el resfriado y para la gripe, si cierto síntoma aparece o no. La tabla fue compuesta a partir del trabajo de especialistas que, conociendo ambas enfermedades, dan cuenta de los síntomas de cada una de ellas.

### 3.3.2. Modelado del Problema

Para modelar nuestro ejemplo, supongamos que tenemos un paciente que inicialmente presenta los siguientes síntomas, no presenta mialgia, ni dolor lumbar, pero presenta cefalea, tos productiva, estornudos, rinorrea, odinofalgia e irritación ocular y después de un lapso de tiempo el paciente presenta sínto-

| Características | Gripe | Resfriado |
|---|---|---|
| inicio | súbito | paulatino |
| mialgia | si | no |
| cefalea | muy intensa | rara |
| tos (productiva) | no | si |
| dolor lumbar | si | no |
| estornudos | raro | si |
| rinorrea | a veces | si |
| odinofalgia | a veces | si |
| irritación ocular | a veces | si |
| secreción nasal acuosa | a veces | si (primeros días) |
| duración | 3-5 días | 8-10 días |
| virus | ortomixovirus | piconavirus y paramixovirus |

Cuadro 3.1: Diferencias entre los síntomas del resfriado y la gripe

mas adicionales tales como dolor lumbar, mialgia, tos seca, fiebre en aumento, cefalea, rinorrea, odinofalgia e irritación ocular. Lo que buscamos es por medio de estos síntomas poder dar un diagnóstico.

Una vez que ya conocemos los síntomas de ambas enfermedades, para modelar el problema lo primero que hicimos es, verificar que síntomas son los síntomas que se presentan en ambas enfermedades y cuáles son los síntomas adicionales o diferentes que nos permitan distinguir la gripe del resfriado.

Basándonos en el cuadro 3.1 podemos observar que los síntomas como cefalea, tos productiva, estornudos, rinorrea, odinofalgia e irritación ocular son síntomas correspondientes al resfriado y los síntomas como dolor lumbar, mialgia, tos seca, fiebre en aumento, cefalea, rinorrea, odinofalgia e irritacion ocular son síntomas correspondientes a la gripe.
Para modelar el problema representamos la información por medio de programas lógicos, dichos programas lógicos se conforman por reglas y restricciones que nos permiten darle un significado a nuestro problema. Utilizaremos 3 programas lógicos que representamos con $P1$, $P2$ y $P3$. El programa $P1$ lo utilizamos para representar la información correspondiente a los síntomas iniciales

que presento el paciente, dicho paciente no presento mialgia, ni dolor lumbar, pero presento cefalea, tos productiva, estornudos, rinorrea, odinofalgia e irritación ocular. El programa $P2$ lo utilizamos para representar la información que indica los síntomas por los cuales el doctor puede diagnosticar resfriado, si el paciente no presenta mialgia ni dolor lumbar, pero presenta cefalea, tos productiva, estornudos, rinorrea, odinofalgia e irritacion ocular es probable que el paciente padezca de resfriado. El programa $P3$ lo utilizamos para representar la información correspondiente a los síntomas adicionales que presento el paciente, los nuevos síntomas del paciente son dolor lumbar, tos seca, mialgia y fiebre en aumento, también colocamos los síntomas por los cuales el doctor puede diferenciar la gripe del resfriado y los síntomas por los cuales el doctor puede diagnosticar gripe. Si el paciente presenta dolor lumbar, no presenta tos productiva, presenta mialgia y fiebre en aumento es un indicio para indicar que el paciente no tiene resfriado, pero si presenta dolor lumbar, mialgia, tos seca, fiebre aumento, cefalea, rinorrea, odinofalgia e irritacion ocular es un indicio que indica que el paciente puede presentar gripe.

Ya que sabemos que información corresponde a cada programa, ahora la colocaremos en forma de reglas y restricciones de manera formal.

Inicialmente en $P1$ colocamos los primeros síntomas que presento el paciente.

$P1$ :

```
r1: -mialgia.
r2: -dolor_lumbar.
r3: cefalea.
r4: tos_productiva.
r5: estornudos.
r6: rinorrea.
r7: odinofalgia.
r8: irritacion_ocular.
```

La regla r1 indica que el paciente no presenta mialgia.

La regla r2 indica que el paciente no presenta dolor lumbar.

La regla r3 indica que el paciente presenta cefalea.

La regla r4 indica que el paciente presenta tos productiva.

La regla r5 indica que el paciente presenta estornudos.
La regla r6 indica que el paciente presenta rinorrea.
La regla r7 indica que el paciente presenta odinofalgia.
La regla r8 indica que el paciente presenta irritación ocular.
El programa P2 representa los síntomas por los cuales un doctor puede diagnósticar resfriado a un paciente.
$P2$ :
`r1: resfrirado ← not mialgia,not dolor_lumbar,cefalea,tos_productiva, estornudos,rinorrea,odinofalgia,irritacion_ocular.`
La regla r1 indica si el paciente no presenta mialgia, no presenta dolor lumbar, presenta cefalea, tos productiva, estornudos, rinorrea, odinofalgia e irritacion ocular entonces tiene resfriado.
El programa P3 representa los nuevos síntomas que presenta el paciente, asi como tambien los sintomas por los cuales se diferencia el resfriado de la gripe y los síntomas por los cuales el doctor puede diagnósticar gripe a un paciente.
$P3$ :
`r1: dolor_lumbar.`
`r2: tos_seca.`
`r3: mialgia.`
`r4: fiebre_aumento.`
`r5: -tos_productiva ← tos_seca.`
`r6: -resfriado ← not tos_productiva, dolor_lumbar , mialgia, fiebre_aumento.`
`r7: gripe ← dolor_lumbar, mialgia, tos_seca, fiebre_aumento, cefalea, rinorrea, odinofalgia, irritacion_ocular.`
La regla r1 indica que el paciente presento dolor lumbar.
La regla r2 indica que el paciente presento tos seca.
La regla r3 indica que el paciente presento mialgia.
La regla r4 indica que el paciente presento fiebre en aumento.
La regla r5 indica que si el paciente no presenta tos productiva, presenta dolor lumbar, mialgia y fiebre en aumento entonces no tiene resfriado.
La regla r6 indica que si el paciente presenta tos seca, entonces no tiene tos

productiva
La regla r7 indica que si el paciente presenta dolor lumbar, mialgia, tos seca, fiebre en aumento, cefalea, rinorrea, odinofalgia e irritacion ocular entonces tiene gripe.
Ahora ya que teniendo los programas, aplicaremos el enfoque de actualización para ver el comportamiento de este ejemplo.

### 3.3.3. Aplicación del Enfoque de Actualización

De manera formal decimos:
Sea $\mathbf{P}$=$P1, P2, P3$ una secuencia de programas, donde $P1, P2, P3$ son los siguientes programas:
$P1$ :

```
r1: -mialgia.
r2: -dolor_lumbar.
r3: cefalea.
r4: tos_productiva.
r5: estornudos.
r6: rinorrea.
r7: odinofalgia.
r8: irritacion_ocular.
```

$P2$ :

```
r1: resfrirado ← not mialgia,not dolor_lumbar,cefalea,tos_productiva,
estornudos,rinorrea,odinofalgia,irritacion_ocular.
```

$P3$ :

```
r1: dolor_lumbar.
r2: tos_seca.
r3: mialgia.
r4: fiebre_aumento.
r5: -tos_productiva   ← tos_seca.
r6: -resfriado   ← not tos_productiva,dolor_lumbar,mialgia,
fiebre_aumento.
```

```
r7: gripe        ← dolor_lumbar, mialgia, tos_seca, fiebre_aumento,
cefalea, rinorrea, odinofalgia, irritacion_ocular.
```

Y sea $B$ el conjunto de nuevos átomos abductivos $\{x_1^1, x_2^1, x_3^1, x_4^1, x_5^1, x_6^1, x_7^1, x_8^1, x_1^2\}$. El nuevo programa lógico de actualización $P\oslash = P1 \oslash P2 \oslash P3$ el siguiente programa:

```
r1: -mialgia           ← not x11.
r2: -dolor_lumbar      ← not x12.
r3: cefalea            ← not x13.
r4: tos_productiva     ← not x14.
r5: estornudos         ← not x15.
r6: rinorrea           ← not x16.
r7: odinofalgia        ← not x17.
r8: irritacion_ocular  ← not x18.
r9: resfriado          ← not mialgia,not dolor_lumbar,cefalea,
tos_productiva,estornudos,rinorrea,odinofalgia,irritacion_ocular,
not x21.
r10: dolor_lumbar.
r11: -tos_productiva   ← tos_seca.
r12: tos_seca.
r13: mialgia.
r14: fiebre_aumento.
r15: -resfriado        ← not tos_productiva, dolor_lumbar,
mialgia,fiebre_aumento.
r16: gripe             ← dolor_lumbar,mialgia,tos_seca,
fiebre_aumento,cefalea,rinorrea,odinofalgia,irritacion_ocular.
```

Ya obtenido el nuevo programa actualizado, ahora obtenemos el conjunto de EEG answer sets.

### 3.3.4. Obtención de los Extendidos Explicitos Generalizados Answer Sets (EEGS)

Una vez obtenido el nuevo programa $P_{\oslash}$, por medio del conjunto potencia de los abductivos y DLV obtenemos el conjunto de EEG answer sets mostrados en el apendice C 4.

### 3.3.5. Obtención de los Mínimos Extendididos Generalizados Answer Sets (MEGS)

Ya teniendo todo el conjunto de EEG aswer sets obtenemos por medio de la inclusion y cardinalidad como se muestra en definición 1.9 el MEG answer set. Por lo tanto, el nevo programa EAL de **P** es un triple $\langle P_{\oslash}, B, f\rangle$ donde $f{:}\rightarrow \{1,2\}$es la función $f(b^i_j = i)$ y el MEG answer sets de $\langle P_{\oslash}, B, f\rangle$ es:
$\{x11, x12, mialgia, dolor\_lumbar, tos\_seca, fiebre\_aumento, cefalea,$
$estornudos, rinorrea, odinofalgia, irritacion\_ocular, -resfriado$
$, gripe\}$. Por lo tanto, el answer set $\oslash$-actualización de **P** es:
$\{mialgia, dolor\_lumbar, tos\_seca, fiebre\_aumento, cefalea, estornudos, rinorrea,$
$odinofalgia, irritacion\_ocular, -resfriado, gripe\}$.

### 3.3.6. Discusión de Resultados

Observando el MEG answer set
$\{mialgia, dolor\_lumbar, tos\_seca, fiebre\_aumento, cefalea, estornudos,$
$rinorrea, odinofalgia, irritacion\_ocular, -resfriado, gripe\}$
nos damos cuenta que la información logro actualizarse ya que inicialmente colocamos síntomas correspondientes al resfriado, pero una vez obtenidos nuevos síntomas la información debia actualizarse y dar una respuesta con respecto a la nueva información. Los síntomas adicionales colocados fueron síntomas intencionales que nos permitierón ver como el sistema despúes de conocer una información logra omitir esa información y actuar mediante las nuevas reglas e información presentada. En este caso la información correspondiente es la

información correspondiente a los síntomas de la gripe y con esta nueva información logramos ver como el sistema omite el indicio de que el paciente tiene resfriado dando lugar a que ahora con nuevos síntomas el paciente no tiene resfriado ahora presenta síntomas correspondientes a la gripe.

### 3.3.7. Conclusiones

Por lo tanto concluimos que el enfoque de actualización es importante en este tipo de procesos donde se necesita actualizar constantemente información y hacer diferentes uso de hipótesis. Lo que buscamos es mostrar como este enfoque de actualización podría ser útil en el modelado de aplicaciones para el diagnóstico de enfermedades ya que este tipo de procesos requieren el uso de razonamientos abductivos y que por medio de semántica de actualización se puede llevar acabo.

# Capítulo 4

# Conclusiones y Trabajos a Futuro

En esta tesis hemos propuesto investigar y evaluar las propiedades del enfoque de actualización de programas lógicos basado en Asnwer Set Programming modelando un conjunto de problemas. Hemos implementado el enfoque de actualización para tres problemas distintos logrando cumplir con los objetivos principales ya que al modelar problemas distintos fuimos capaces de estudiar y entender la potencialidad del enfoque de actualización. Parte fundamental de esta tesis támbien fue la investigación de las características del enfoque de actualización así como también de los temas relacionados. Los problemas que presentamos se llevaron acabo siguiendo la sémantica del enfoque de actualización observando que en los tres problemas obtuvimos resultados esperados por lo tanto llegamos a la conclusión que respecto al enfoque de actualización no solo podemos observar la utilidad que tiene en un agente inteligente, sino en cualquier actividad o área del conocimiento donde halla la necesidad actualización.

Concluimos que el gran potencial de este enfoque de actualización es su capacidad de tratar con problemas de información incompleta, información cambiante o información contradictoria y gracias a estas características es posible resolver problemas complejos de una manera sencilla.

Finalmente, proponemos ampliar el estudio de este enfoque para encontrar otras variantes de la semántica de answer sets y así analizar otras situaciones ya que existen muchos dominios en donde podría aplicarse actualización. Proponemos como trabajos a futuro desarrollar una aplicación específica para poner en práctica el enfoque de actualización que debido al tiempo no fue posible realizarla.

# Apéndice A

Extendidos Explicitos Generalizados Answer Sets correspondientes al problema 3.1.

| *numero* Δ | Δ | ⟨*M*,Δ⟩ |
|---|---|---|
| 1 | { } | No tiene EEG answer sets |
| 2 | {x11} | {x11,tanque1_sin_agua, cisterna_vacia,lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 3 | {x21} | No tiene EEG answer sets |
| 4 | {x11,x21} | {x11, tanque1_sin_agua, x21, cisterna_vacia, -activar_bomba, -lampara3_encendida} |
| 5 | {x22} | No tiene EEG answer sets |
| 6 | {x11,x22} | {x11, tanque1_sin_agua, x22, cisterna_vacia, lampara1_encendida, -activar_bomba,-lampara3_encendida} |
| 7 | {x21,x22} | No tiene EEG answer sets |
| 8 | {x11,x21,x22} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, -activar_bomba, -lampara3_encendida} |
| 9 | {x23} | No tiene EEG answer sets |
| 10 | {x11,x23} | {x11, tanque1_sin_agua, cisterna_vacia, x23, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 11 | {x21,x23} | No tiene EEG answer sets |
| 12 | {x11,x21,x23} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x23, -activar_bomba, -lampara3_encendida} |

| *numero* Δ | Δ | ⟨*M*,Δ⟩ |
|---|---|---|
| 13 | {x22,x23} | No tiene EEG answer sets |
| 14 | {x11,x22,x23} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x23, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 15 | {x21,x22,x23} | No tiene EEG answer sets |
| 16 | {x11,x21,x22,x23} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x23, -activar_bomba, -lampara3_encendida} |
| 17 | {x31} | No tiene EEG answer sets |
| 18 | {x11,x31} | {x11, tanque1_sin_agua, cisterna_vacia, x31, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 19 | {x21,x31} | No tiene EEG answer sets |
| 20 | {x11,x21,x31} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x31, -activar_bomba, -lampara3_encendida} |
| 21 | {x22,x31} | No tiene EEG answer sets |
| 22 | {x11,x22,x31} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x31, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 23 | {x21,x22,x31} | No tiene EEG answer sets |
| 24 | {x11,x21,x22,x31} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x31, -activar_bomba, -lampara3_encendida} |
| 25 | {x23,x31} | No tiene EEG answer sets |

| | | |
|---|---|---|
| 26 | {x11,x23,x31} | {x11, tanque1_sin_agua, cisterna_vacia, x23, x31, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 27 | {x21,x23,x31} | No tiene EEG answer sets |
| 28 | {x11,x21,x23,x31} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x23, x31, -activar_bomba, -lampara3_encendida} |
| 29 | {x22,x23,x31} | No tiene EEG answer sets |
| 30 | {x11,x22,x23,x31} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x23, x31, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 31 | {x21,x22,x23,x31} | No tiene EEG answer sets |
| 32 | {x11,x21,x22,x23,x31} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x23, x31, -activar_bomba, -lampara3_encendida} |
| 33 | {x32} | No tiene EEG answer sets |
| 34 | {x11,x32} | {x11, tanque1_sin_agua, cisterna_vacia, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 35 | {x21,x32} | No tiene EEG answer sets |
| 36 | {x11,x21,x32} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x32, -activar_bomba, -lampara3_encendida} |
| 37 | {x22,x32} | No tiene EEG answer sets |
| 38 | {x11,x22,x32} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |

| | | |
|---|---|---|
| 39 | {x21,x22,x32} | No tiene EEG answer sets |
| 40 | {x11,x21,x22,x32} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x32, -activar_bomba, -lampara3_encendida} |
| 41 | {x23,x32} | No tiene EEG answer sets |
| 42 | {x11,x23,x32} | {x11, tanque1_sin_agua, cisterna_vacia, x23, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 43 | {x21,x23,x32} | No tiene EEG answer sets |
| 44 | {x11,x21,x23,x32} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x23, x32, -activar_bomba, -lampara3_encendida} |
| 45 | {x22,x23,x32} | No tiene EEG answer sets |
| 46 | {x11,x22,x23,x32} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x23, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 47 | {x21,x22,x23,x32} | No tiene EEG answer sets |
| 48 | {x11,x21,x22,x23,x32} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x23, x32, -activar_bomba, -lampara3_encendida} |
| 49 | {x31,x32} | No tiene EEG answer sets |
| 50 | {x11,x31,x32} | {x11, tanque1_sin_agua, cisterna_vacia, x31, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 51 | {x21,x31,x32} | No tiene EEG answer sets |
| 52 | {x11,x21,x31,x32} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x31, x32, -activar_bomba, -lampara3_encendida} |

| | | |
|---|---|---|
| 53 | {x22,x31,x32} | No tiene EEG answer sets |
| 54 | {x11,x22,x31,x32} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x31, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 55 | {x21,x22,x31,x32} | No tiene EEG answer sets |
| 56 | {x11,x21,x22,x31,x32} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x31, x32, -activar_bomba, -lampara3_encendida} |
| 57 | {x23,x31,x32} | No tiene EEG answer sets |
| 58 | {x11,x23,x31,x32} | {x11, tanque1_sin_agua, cisterna_vacia, x23, x31, x32, lampara1_encendida, -activar_bomba, -lampara3_encendida} |
| 59 | {x21,x23,x31,x32} | No tiene EEG answer sets |
| 60 | {x11,x21,x23,x31,x32} | {x11, tanque1_sin_agua, x21, cisterna_vacia, x23, x31, x32, -activar_bomba, -lampara3_encendida} |
| 61 | {x22,x23,x31,x32} | No tiene EEG answer sets |
| 62 | {x11,x22,x23,x31,x32} | {x11, tanque1_sin_agua, x22, cisterna_vacia, x23, x31, x32, lampara1_encendida,-activar_bomba, -lampara3_encendida} |
| 63 | {x21,x22,x23,x31,x32} | No tiene EEG answer sets |
| 64 | {x11,x21,x22,x23,x31,x32} | {x11, tanque1_sin_agua, x21, x22, cisterna_vacia, x23, x31, x32, -activar_bomba,-lampara3_encendida} |

# Apéndice B

Extendidos Explicitos Generalizados Answer Sets correspondientes al problema 3.2.

| *numero* Δ | Δ | ⟨*M*,Δ⟩ |
|---|---|---|
| 1 | { } | No tiene EEG answer sets |
| 2 | {x11} | No tiene EEG answer sets |
| 3 | {x12} | No tiene EEG answer sets |
| 4 | {x11,x12} | No tiene EEG answer sets |
| 5 | {x13} | {x13, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 6 | {x11,x13} | {x11, x13, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 7 | {x12,x13} | {x12, x13, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 8 | {x11,x12,x13} | {x11, x12, x13, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 9 | {x14} | No tiene EEG answer sets |
| 10 | {x11,x14} | No tiene EEG answer sets |
| 11 | {x12,x14} | No tiene EEG answer sets |
| 12 | {x11,x12,x14} | No tiene EEG answer sets |
| 13 | {x13,x14} | {x13, x14, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 14 | {x11,x13,x14} | {x11, x13, x14, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 15 | {x12,x13,x14} | {x12, x13, x14, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 16 | {x11,x12,x13,x14} | {x11, x12, x13, x14, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |

| | | |
|---|---|---|
| 17 | {x21} | No tiene EEG answer sets |
| 18 | {x11,x21} | No tiene EEG answer sets |
| 19 | {x12,x21} | No tiene EEG answer sets |
| 20 | {x11,12,x21} | No tiene EEG answer sets |
| 21 | {x13,x21} | {x13, x21, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 22 | {x11,x13,x21} | {x11, x13, x21, sensor3_detecta_precencia, -en_lamp2, -en_lamp4, usuario_atrasD,en_lamp3} |
| 23 | {x12,x13,x21} | {x12, x13, x21, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 24 | {x11,x12,x13,x21} | {x11, x12, x13, x21, sensor3_detecta_precencia, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 25 | {x14,x21} | No tiene EEG answer sets |
| 26 | {x11,x14,x21} | No tiene EEG answer sets |
| 27 | {x12,x14,x21} | No tiene EEG answer sets |
| 28 | {x11,x12,x14,x21} | No tiene EEG answer sets |
| 29 | {x13,x14,x21} | {x13, x14, x21, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 30 | {x11,x13,x14,x21} | {x11, x13, x14, x21, sensor3_detecta_precencia, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 31 | {x12,x13,x14,x21} | {x12, x13, x14, x21, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 32 | {x11,x12,x13,x14,x21} | {x11, x12, x13, x14, x21, sensor3_detecta_precencia, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 33 | {x22} | No tiene EEG answer sets |
| 34 | {x11,x22} | No tiene EEG answer sets |
| 35 | {x12,x22} | No tiene EEG answer sets |
| 36 | {x11,x12,x22} | No tiene EEG answer sets |
| 37 | {x13,x22} | {x13, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 38 | {x11,x13,x22} | {x11, x13, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 39 | {x12,x13,x22} | {x12, x13, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 40 | {x11,x12,x13,x22} | {x11, x12, x13, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 41 | {x14,x22} | No tiene EEG answer sets |
| 42 | {x11,x14,x22} | No tiene EEG answer sets |
| 43 | {x12,x14,x22} | No tiene EEG answer sets |
| 44 | {x11,x12,x14,x22} | No tiene EEG answer sets |
| 45 | {x13,x14,x22} | {x13, x14, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 46 | {x11,x13,x14,x22} | {x11, x13, x14, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 47 | {x12,x13,x14,x22} | {x12, x13, x14, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 48 | {x11,x12,x13,x14,x22} | {x11, x12, x13, x14, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 49 | {x21,x22} | No tiene EEG answer sets |
| 50 | {x11,x21,x22} | No tiene EEG answer sets |
| 51 | {x12,x21,x22} | No tiene EEG answer sets |
| 52 | {x11,x12,x21,x22} | No tiene EEG answer sets |
| 53 | {x13,x21,x22} | {x13, x21, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 54 | {x11,x13,x21,x22} | {x11, x13, x21, x22, sensor3_detecta_precencia, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 55 | {x12,x13,x21,x22} | {x12, x13, x21, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |

| | | |
|---|---|---|
| 56 | {x11,x12,x13,x21,x22} | {x11, x12, x13, x21, x22, sensor3_detecta_precencia, -en_lamp4, usuario_atrasD,en_lamp3} |
| 57 | {x14,x21,x22} | No tiene EEG answer sets |
| 58 | {x11,x14,x21,x22} | No tiene EEG answer sets |
| 59 | {x12,x14,x21,x22} | No tiene EEG answer sets |
| 60 | {x11,x12,x14,x21,x22} | No tiene EEG answer sets |
| 61 | {x13,x14,x21,x22} | {x13, x14, x21, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 62 | {x11,x13,x14,x21,x22} | {x13, x14, x21, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 63 | {x12,x13,x14,x21,x22} | {x12, x13, x14, x21, x22, sensor3_detecta_precencia, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 64 | {x11,x12,x13,x14,x21x22} | {x11, x12, x13, x14, x21, x22, sensor3_detecta_precencia, -en_lamp4, usuario_atrasD, en_lamp3} |
| 65 | {x23} | No tiene EEG answer sets |
| 66 | {x11,x23} | No tiene EEG answer sets |
| 67 | {x12,x23} | No tiene EEG answer sets |
| 68 | {x11,x12,x23} | No tiene EEG answer sets |
| 69 | {x13,x23} | {x13, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 70 | {x11,x13,x23} | {x11, x13, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 71 | {x12,x13,x23} | {x12, x13, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 72 | {x11,x12,x13,x23} | {x11, x12, x13, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 73 | {x14,x23} | No tiene EEG answer sets |
| 74 | {x11,x14,x23} | No tiene EEG answer sets |
| 75 | {x12,x14,x23} | No tiene EEG answer sets |
| 76 | {x11,x12,x14,x23} | No tiene EEG answer sets |
| 77 | {x13,x14,x23} | {x13, x14, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 78 | {x11,x13,x14,x23} | {x11, x13, x14, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 79 | {x12,x13,x14,x23} | {x12, x13, x14, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 80 | {x11,x12,x13,x14,x23} | {x11, x12, x13, x14, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 81 | {x21,x23} | No tiene EEG answer sets |
| 82 | {x11,x21,x23} | No tiene EEG answer sets |
| 83 | {x12,x21,x23} | No tiene EEG answer sets |
| 84 | {x11,x12,x21,x23} | No tiene EEG answer sets |
| 85 | {x13,x21,x23} | {x13, x21, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 86 | {x11,x13,x21,x23} | {x11, x13, x21, sensor3_detecta_precencia, x23, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 87 | {x12,x13,x21,x23} | {x12, x13, x21, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 88 | {x11,x12,x13,x21,x23} | {x11, x12, x13, x21, sensor3_detecta_precencia, x23, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 89 | {x14,x21,x23} | No tiene EEG answer sets |
| 90 | {x11,x14,x21,x23} | No tiene EEG answer sets |

| | | |
|---|---|---|
| 91 | {x12,x14,x21,x23} | No tiene EEG answer sets |
| 92 | {x11,x12,x14,x21,x23} | No tiene EEG answer sets |
| 93 | {x13,x14,x21,x23} | {x13, x14, x21, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 94 | {x11,x13,x14,x21,x23} | {x11, x13, x14, x21, sensor3_detecta_precencia, x23, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 95 | {x12,x13,x14,x21,x23} | {x12, x13, x14, x21, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 96 | {x11,x12,x13,x14,x21,x23} | {x11, x12, x13, x14, x21, sensor3_detecta_precencia, x23, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 97 | {x22,x23} | No tiene EEG answer sets |
| 98 | {x11,x22,x23} | No tiene EEG answer sets |
| 99 | {x12,x22,x23} | No tiene EEG answer sets |
| 100 | {x11,x12,x22,x23} | No tiene EEG answer sets |
| 101 | {x13,x22,x23} | {x13, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 102 | {x11,x13,x22,x23} | {x11, x13, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 103 | {x12,x13,x22,x23} | {x12, x13, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 104 | {x11,x12,x13,x22,x23} | {x11, x12, x13, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 105 | {x14,x22,x23} | No tiene EEG answer sets |
| 106 | {x11,x14,x22,x23} | No tiene EEG answer sets |

| | | |
|---|---|---|
| 107 | {x12,x14,x22,x23} | No tiene EEG answer sets |
| 108 | {x11,x12,x14,x22,x23} | No tiene EEG answer sets |
| 109 | {x13,x14,x22,x23} | {x13, x14, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 110 | {x11,x13,x14,x22,x23} | {x11, x13, x14, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 111 | {x12,x13,x14,x22,x23} | {x12, x13, x14, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 112 | {x11,x12,x13,x14,x22,x23} | {x11, x12, x13, x14, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp4,usuario_atrasD, en_lamp3} |
| 113 | {x21,x22,x23} | No tiene EEG answer sets |
| 114 | {x11,x21,x22,x23} | No tiene EEG answer sets |
| 115 | {x12,x21,x22,x23} | No tiene EEG answer sets |
| 116 | {x11,x12,x21,x22,x23} | No tiene EEG answer sets |
| 117 | {x13,x21,x22,x23} | {x13, x21, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 118 | {x11,x13,x21,x22,x23} | {x11, x13, x21, x22, sensor3_detecta_precencia, x23, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 119 | {x12,x13,x21,x22,x23} | {x12, x13, x21, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 120 | {x11,x12,x13,x21,x22,x23} | {x11, x12, x13, x21, x22, sensor3_detecta_precencia, x23, -en_lamp4, usuario_atrasD, en_lamp3} |
| 121 | {x14,x21,x22,x23} | No tiene EEG answer sets |

| | | |
|---|---|---|
| 122 | {x11,x14,x21,x22,x23} | No tiene EEG answer sets |
| 123 | {x12,x14,x21,x22,x23} | No tiene EEG answer sets |
| 124 | {x11,x12,x14,x21,x22,x23} | No tiene EEG answer sets |
| 125 | {x13,x14,x21,x22,x23} | {x13, x14, x21, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 126 | {x11,x13,x14,x21,x2223} | {x11, x13, x14, x21, x22, sensor3_detecta_precencia, x23, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 127 | {x12,x13,x14,x21,x22,x23} | {x12, x13, x14, x21, x22, sensor3_detecta_precencia, x23, -en_lamp1, -en_lamp4,usuario_atrasD, en_lamp3} |
| 128 | {x11,x12,x13,x14,x21,x22,x23} | {x11, x12, x13, x14, x21, x22, sensor3_detecta_precencia, x23, -en_lamp4, usuario_atrasD, en_lamp3} |
| 129 | {x24} | No tiene EEG answer sets |
| 130 | {x11,x24} | No tiene EEG answer sets |
| 131 | {x12,x24} | No tiene EEG answer sets |
| 132 | {x11,x12,x24} | No tiene EEG answer sets |
| 133 | {x13,x24} | {x13, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 134 | {x11,x13,x24} | {x11, x13, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 135 | {x12,x13,x24} | {x12, x13, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 136 | {x11,x12,x13,x24} | {x11, x12, x13, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 137 | {x14,x24} | No tiene EEG answer sets |
| 138 | {x11,x14,x24} | No tiene EEG answer sets |
| 139 | {x12,x14,x24} | No tiene EEG answer sets |

| | | |
|---|---|---|
| 140 | {x11,x12,x14,x24} | No tiene EEG answer sets |
| 141 | {x13,x14,x24} | {x13, x14, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD,en_lamp3} |
| 142 | {x11,x13,x14,x24} | {x11, x13, x14, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 143 | {x12,x13,x14,x24} | {x12, x13, x14, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 144 | {x11,x12,x13,x14,x24} | {x11, x12, x13, x14, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 145 | {x21,x24} | No tiene EEG answer sets |
| 146 | {x11,x21,x24} | No tiene EEG answer sets |
| 147 | {x12,x21,x24} | No tiene EEG answer sets |
| 148 | {x11,x12,x21,x24} | No tiene EEG answer sets |
| 149 | {x13,x21,x24} | {x13, x21, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 150 | {x11,x13,x21,x24} | {x11, x13, x21, sensor3_detecta_precencia, x24, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 151 | {x12,x13,x21,x24} | {x12, x13, x21, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 152 | {x11,x12,x13,x21,x24} | {x11, x12, x13, x21, sensor3_detecta_precencia, x24, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 153 | {x14,x21,x24} | No tiene EEG answer sets |
| 154 | {x11,x14,x21,x24} | No tiene EEG answer sets |
| 155 | {x12,x14,x21,x24} | No tiene EEG answer sets |
| 156 | {x11,x12,x14,x21,x24} | No tiene EEG answer sets |
| 157 | {x13,x14,x21,x24} | {x13, x14, x21, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |

| | | |
|---|---|---|
| 158 | {x11,x13,x14,x21,x24} | {x11, x13, x14, x21, sensor3_detecta_precencia, x24, -en_lamp2, usuario_atrasD,en_lamp3} |
| 159 | {x12,x13,x14,x21,x24} | {x12, x13, x14, x21, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 160 | {x11,x12,x13,x14,x21,x24} | {x11, x12, x13, x14, x21, sensor3_detecta_precencia, x24, -en_lamp2, usuario_atrasD, en_lamp3} |
| 161 | {2224} | No tiene EEG answer sets |
| 162 | {112224} | No tiene EEG answer sets |
| 163 | {122224} | No tiene EEG answer sets |
| 164 | {11122224} | No tiene EEG answer sets |
| 165 | {132224} | {x13, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 166 | {11132224} | {x11, x13, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 167 | {12132224} | {x12, x13, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 168 | {1112132224} | {x11, x12, x13, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 169 | {142224} | No tiene EEG answer sets |
| 170 | {11142224} | No tiene EEG answer sets |
| 171 | {12142224} | No tiene EEG answer sets |
| 172 | {1112142224} | No tiene EEG answer sets |
| 173 | {13142224} | {x13, x14, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 174 | {1113142224} | {x11, x13, x14, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 175 | {1213142224} | {x12, x13, x14, x22, sensor3_detecta_precencia, x24, -en_lamp1, usuario_atrasD,en_lamp3} |
| 176 | {111213142224} | {x11, x12, x13, x14, x22, sensor3_detecta_precencia, x24, -en_lamp1, usuario_atrasD, en_lamp3} |
| 177 | {212224} | No tiene EEG answer sets |
| 178 | {11212224} | No tiene EEG answer sets |
| 179 | {12212224} | No tiene EEG answer sets |
| 180 | {1112212224} | No tiene EEG answer sets |
| 181 | {13212224} | {x13, x21, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 182 | {1113212224} | {x11, x13, x21, x22, sensor3_detecta_precencia, x24, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 183 | {1213212224} | {x12, x13, x21, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 184 | {111213212224} | {x11, x12, x13, x21, x22, sensor3_detecta_precencia, x24, -en_lamp4, usuario_atrasD, en_lamp3} |
| 185 | {14212224} | No tiene EEG answer sets |
| 186 | {1114212224} | No tiene EEG answer sets |
| 187 | {1214212224} | No tiene EEG answer sets |
| 188 | {111214212224} | No tiene EEG answer sets |
| 189 | {1314212224} | {x13, x14, x21, x22, sensor3_detecta_precencia, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 190 | {111314212224} | {x11, x13, x14, x21, x22, sensor3_detecta_precencia, x24, -en_lamp2, usuario_atrasD, en_lamp3} |
| 191 | {121314212224} | {x12, x13, x14, x21, x22, sensor3_detecta_precencia, x24, -en_lamp1, usuario_atrasD, en_lamp3} |
| 192 | {1112131421222 4} | {x11, x12, x13, x14, x21, x22, sensor3_detecta_precencia, x24, usuario_atrasD, en_lamp3} |
| 193 | {2324} | No tiene EEG answer sets |
| 194 | {112324} | No tiene EEG answer sets |

| | | |
|---|---|---|
| 195 | {122324} | No tiene EEG answer sets |
| 196 | {11122324} | No tiene EEG answer sets |
| 197 | {132324} | {x13, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 198 | {11132324} | {x11, x13, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 199 | {12132324} | {x12, x13, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 200 | {1112132324} | {x11, x12, x13, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 201 | {142324} | No tiene EEG answer sets |
| 202 | {11142324} | No tiene EEG answer sets |
| 203 | {12142324} | No tiene EEG answer sets |
| 204 | {1112142324} | No tiene EEG answer sets |
| 205 | {13142324} | {x13, x14, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 206 | {1113142324} | {x11, x13, x14, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 207 | {1213142324} | {x12, x13, x14, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 208 | {111213142324} | {x11, x12, x13, x14, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2,usuario_atrasD, en_lamp3} |
| 209 | {212324} | No tiene EEG answer sets |
| 210 | {11212324} | No tiene EEG answer sets |
| 211 | {12212324} | No tiene EEG answer sets |
| 212 | {1112212324} | No tiene EEG answer sets |
| 213 | {13212324} | {x13, x21, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 214 | {1113212324} | {x11, x13, x21, sensor3_detecta_precencia, x23, x24, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 215 | {1213212324} | {x12, x13, x21, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 216 | {111213212324} | {x11, x12, x13, x21, sensor3_detecta_precencia, x23, x24, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 217 | {14212324} | No tiene EEG answer sets |
| 218 | {1114212324} | No tiene EEG answer sets |
| 219 | {1214212324} | No tiene EEG answer sets |
| 220 | {111214212324} | No tiene EEG answer sets |
| 221 | {1314212324} | {x13, x14, x21, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 222 | {111314212324} | {x11, x13, x14, x21, sensor3_detecta_precencia, x23, x24, -en_lamp2, usuario_atrasD, en_lamp3} |
| 223 | {121314212324} | {x12, x13, x14, x21, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2,usuario_atrasD, en_lamp3} |
| 224 | {1112131421232 4} | {x11, x12, x13, x14, x21, sensor3_detecta_precencia, x23, x24, -en_lamp2, usuario_atrasD, en_lamp3} |
| 225 | {222324} | No tiene EEG answer sets |
| 226 | {11222324} | No tiene EEG answer sets |
| 227 | {12222324} | No tiene EEG answer sets |
| 228 | {1112222324} | No tiene EEG answer sets |
| 229 | {13222324} | {x13, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 230 | {1113222324} | {x11, x13, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |

| | | |
|---|---|---|
| 231 | {1213222324} | {x12, x13, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp4, usuario_atrasD, en_lamp3} |
| 232 | {111213222324} | {x11, x12, x13, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp4,usuario_atrasD, en_lamp3} |
| 233 | {14222324} | No tiene EEG answer sets |
| 234 | {1114222324} | No tiene EEG answer sets |
| 235 | {1214222324} | No tiene EEG answer sets |
| 236 | {111214222324} | No tiene EEG answer sets |
| 237 | {1314222324} | {x13, x14, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, usuario_atrasD, en_lamp3} |
| 238 | {111314222324} | {x11, x13, x14, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2,usuario_atrasD, en_lamp3} |
| 239 | {121314222324} | {x12, x13, x14, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, usuario_atrasD, en_lamp3} |
| 240 | {1112131422232 4} | {x11, x12, x13, x14, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, usuario_atrasD, en_lamp3} |
| 241 | {21222324} | No tiene EEG answer sets |
| 242 | {1121222324} | No tiene EEG answer sets |
| 243 | {1221222324} | No tiene EEG answer sets |
| 244 | {111221222324} | No tiene EEG answer sets |
| 245 | {1321222324} | {x13, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2, -en_lamp4, usuario_atrasD, en_lamp3} |
| 246 | {111321222324} | {x11, x13, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp2, -en_lamp4,usuario_atrasD, en_lamp3} |
| 247 | {121321222324} | {x12, x13, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp4,usuario_atrasD, en_lamp3} |
| 248 | {1112132122232 4} | {x11, x12, x13, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp4, usuario_atrasD, en_lamp3} |
| 249 | {1421222324} | No tiene EEG answer sets |
| 250 | {111421222324} | No tiene EEG answer sets |
| 251 | {121421222324} | No tiene EEG answer sets |
| 252 | {1112142122232 4} | No tiene EEG answer sets |
| 253 | {131421222324} | {x13, x14, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, -en_lamp2,usuario_atrasD, en_lamp3} |
| 254 | {1113142122232 4} | {x11, x13, x14, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp2, usuario_atrasD, en_lamp3} |
| 255 | {1213142122232 4} | {x12, x13, x14, x21, x22, sensor3_detecta_precencia, x23, x24, -en_lamp1, usuario_atrasD, en_lamp3} |
| 256 | {1112131421222 324} | {x11, x12, x13, x14, x21, x22, sensor3_detecta_precencia, x23, x24, usuario_atrasD, en_lamp3} |

# Apéndice C

Extendidos Explicitos Generalizados Answer Sets correspondientes al problema 3.3.

| *numero* Δ | Δ | ⟨*M*, Δ⟩ |
|---|---|---|
| 1 | { } | No EEG Answer sets |
| 2 | {x11} | No EEG Answer sets |
| 3 | {x12} | No EEG Answer sets |
| 4 | {x11,x12} | {x11, x12, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, odinofalgia, irritacion_ocular, -resfriado, gripe} |
| 5 | {x13} | No EEG Answer sets |
| 6 | {x11,x13} | No EEG Answer sets |
| 7 | {x12,x13} | No EEG Answer sets |
| 8 | {x11,x12,x13} | {x11, x12, x13, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, rinorrea, odinofalgia, irritacion_ocular, -resfriado} |
| 9 | {x14} | No EEG Answer sets |
| 10 | {x11,x14} | No EEG Answer sets |
| 11 | {x12,x14} | No EEG Answer sets |
| 12 | {x11,x12,x14} | {x11, x12, x14, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, rinorrea, odinofalgia, irritacion_ocular, -resfriado, gripe} |
| 13 | {x13,x14} | No EEG Answer sets |
| 14 | {x11,x13,x14} | No EEG Answer sets |
| 15 | {x12,x13,x14} | No EEG Answer sets |
| 16 | {x11,x12,x13,x14} | {x11, x12, x13, x14, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, rinorrea,odinofalgia, irritacion_ocular, -resfriado} |
| 17 | {x15} | No EEG Answer sets |
| 18 | {x11,x15} | No EEG Answer sets |
| 19 | {x12,x15} | No EEG Answer sets |
| 20 | {x11,x12,x15} | {x11, x12, x15, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, odinofalgia, irritacion_ocular, -resfriado} |

| | | |
|---|---|---|
| 21 | {x13,x15} | No EEG Answer sets |
| 22 | {x11,x13,x15} | No EEG Answer sets |
| 23 | {x12,x13,x15} | No EEG Answer sets |
| 24 | {x11,x12,x13,x15} | {x11, x12, x13, x15, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, odinofalgia, irritacion_ocular, -resfriado} |
| 25 | {x14,x15} | No EEG Answer sets |
| 26 | {x11,x14,x15} | No EEG Answer sets |
| 27 | {x12,x14,x15} | No EEG Answer sets |
| 28 | {x11,x12,x14,x15} | {x11, x12, x14, x15, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, odinofalgia, irritacion_ocular, -resfriado} |
| 29 | {x13,x14,x15} | No EEG Answer sets |
| 30 | {x11,x13,x14,x15} | No EEG Answer sets |
| 31 | {x12,x13,x14,x15} | No EEG Answer sets |
| 32 | {x11,x12,x13,x14,x15} | {x11, x12, x13, x14, x15, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, odinofalgia, irritacion_ocular, -resfriado} |
| 33 | {x16} | No EEG Answer sets |
| 34 | {x11,x16} | No EEG Answer sets |
| 35 | {x12,x16} | No EEG Answer sets |
| 36 | {x11,x12,x16} | {x11, x12, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, irritacion_ocular, -resfriado} |
| 37 | {x13,x16} | No EEG Answer sets |
| 38 | {x11,x13,x16} | No EEG Answer sets |
| 39 | {x12,x13,x16} | No EEG Answer sets |
| 40 | {x11,x12,x13,x16} | {x11, x12, x13, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, rinorrea, irritacion_ocular, -resfriado} |
| 41 | {x14,x16} | No EEG Answer sets |
| 42 | {x11,x14,x16} | No EEG Answer sets |
| 43 | {x12,x14,x16} | No EEG Answer sets |
| 44 | {x11,x12,x14,x16} | {x11, x12, x14, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, rinorrea, irritacion_ocular, -resfriado} |
| 45 | {x13,x14,x16} | No EEG Answer sets |
| 46 | {x11,x13,x14,x16} | No EEG Answer sets |
| 47 | {x12,x13,x14,x16} | No EEG Answer sets |
| 48 | {x11,x12,x13,x14,x16} | {x11, x12, x13, x14, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, rinorrea, irritacion_ocular, -resfriado} |
| 49 | {x15,x16} | No EEG Answer sets |
| 50 | {x11,x15,x16} | No EEG Answer sets |
| 51 | {x12,x15,x16} | No EEG Answer sets |
| 52 | {x11,x12,x15,x16} | {x11, x12, x15, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, irritacion_ocular, -resfriado} |
| 53 | {x13,x15,x16} | No EEG Answer sets |
| 54 | {x11,x13,x15,x16} | No EEG Answer sets |
| 55 | {x12,x13,x15,x16} | No EEG Answer sets |
| 56 | {x11,x12,x13,x15,x16} | {x11, x12, x13, x15, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, irritacion_ocular, -resfriado} |

| | | |
|---|---|---|
| 57 | {x14,x15,x16} | No EEG Answer sets |
| 58 | {x11,x14,x15,x16} | No EEG Answer sets |
| 59 | {x12,x14,x15,x16} | No EEG Answer sets |
| 60 | {x11,x12,x14,x15,x16} | {x11, x12, x14, x15, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, irritacion_ocular, -resfriado} |
| 61 | {x13,x14,x15,x16} | No EEG Answer sets |
| 62 | {x11,x13,x14,x15,x16} | No EEG Answer sets |
| 63 | {x12,x13,x14,x15,x16} | No EEG Answer sets |
| 64 | {x11,x12,x13,x14,x15,x16} | {x11, x12, x13, x14, x15, x16, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, irritacion_ocular, -resfriado} |
| 65 | {x17} | No EEG Answer sets |
| 66 | {x11,x17} | No EEG Answer sets |
| 67 | {x12,x17} | No EEG Answer sets |
| 68 | {x11,x12,x17} | {x11, x12, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, odinofalgia, -resfriado} |
| 69 | {x13,x17} | No EEG Answer sets |
| 70 | {x11,x13,x17} | No EEG Answer sets |
| 71 | {x12,x13,x17} | No EEG Answer sets |
| 72 | {x11,x12,x13,x17} | {x11, x12, x13, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, rinorrea, odinofalgia, -resfriado} |
| 73 | {x14,x17} | No EEG Answer sets |
| 74 | {x11,x14,x17} | No EEG Answer sets |
| 75 | {x12,x14,x17} | No EEG Answer sets |
| 76 | {x11,x12,x14,x17} | {x11, x12, x14, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, rinorrea, odinofalgia, -resfriado} |
| 77 | {x13,x14,x17} | No EEG Answer sets |
| 78 | {x11,x13,x14,x17} | No EEG Answer sets |
| 79 | {x12,x13,x14,x17} | No EEG Answer sets |
| 80 | {x11,x12,x13,x14,x17} | {x11, x12, x13, x14, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, rinorrea, odinofalgia, -resfriado} |
| 81 | {x15,x17} | No EEG Answer sets |
| 82 | {x11,x15,x17} | No EEG Answer sets |
| 83 | {x12,x15,x17} | No EEG Answer sets |
| 84 | {x11,x12,x15,x17} | {x11, x12, x15, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, odinofalgia, -resfriado} |
| 85 | {x13,x15,x17} | No EEG Answer sets |
| 86 | {x11,x13,x15,x17} | No EEG Answer sets |
| 87 | {x12,x13,x15,x17} | No EEG Answer sets |
| 88 | {x11,x12,x13,x15,x17} | {x11, x12, x13, x15, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, odinofalgia, -resfriado} |
| 89 | {x14,x15,x17} | No EEG Answer sets |
| 90 | {x11,x14,x15,x17} | No EEG Answer sets |

| | | |
|---|---|---|
| 91 | {x12,x14,x15,x17} | No EEG Answer sets |
| 92 | {x11,x12,x14,x15,x17} | {x11, x12, x14, x15, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, odinofalgia, -resfriado} |
| 93 | {x13,x14,x15,x17} | No EEG Answer sets |
| 94 | {x11,x13,x14,x15,x17} | No EEG Answer sets |
| 95 | {x12,x13,x14,x15,x17} | No EEG Answer sets |
| 96 | {x11,x12,x13,x14,x15,x17} | {x11, x12, x13, x14, x15, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, odinofalgia, -resfriado} |
| 97 | {x16,x17} | No EEG Answer sets |
| 98 | {x11,x16,x17} | No EEG Answer sets |
| 99 | {x12,x16,x17} | No EEG Answer sets |
| 100 | {x11,x12,x16,x17} | {x11, x12, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, -resfriado} |
| 101 | {x13,x16,x17} | No EEG Answer sets |
| 102 | {x11,x13,x16,x17} | No EEG Answer sets |
| 103 | {x12,x13,x16,x17} | No EEG Answer sets |
| 104 | {x11,x12,x13,x16,x17} | {x11, x12, x13, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, rinorrea, -resfriado} |
| 105 | {x14,x16,x17} | No EEG Answer sets |
| 106 | {x11,x14,x16,x17} | No EEG Answer sets |
| 107 | {x12,x14,x16,x17} | No EEG Answer sets |
| 108 | {x11,x12,x14,x16,x17} | {x11, x12, x14, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, rinorrea, -resfriado} |
| 109 | {x13,x14,x16,x17} | No EEG Answer sets |
| 110 | {x11,x13,x14,x16,x17} | No EEG Answer sets |
| 111 | {x12,x13,x14,x16,x17} | No EEG Answer sets |
| 112 | {x11,x12,x13,x14,x16,x17} | {x11, x12, x13, x14, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, rinorrea, -resfriado} |
| 113 | {x15,x16,x17} | No EEG Answer sets |
| 114 | {x11,x15,x16,x17} | No EEG Answer sets |
| 115 | {x12,x15,x16,x17} | No EEG Answer sets |
| 116 | {x11,x12,x15,x16,x17} | {x11, x12, x15, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, estornudos, -resfriado} |
| 117 | {x13,x15,x16,x17} | No EEG Answer sets |
| 118 | {x11,x13,x15,x16,x17} | No EEG Answer sets |
| 119 | {x12,x13,x15,x16,x17} | No EEG Answer sets |
| 120 | {x11,x12,x13,x15,x16,x17} | {x11, x12, x13, x15, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, estornudos, -resfriado} |
| 121 | {x14,x15,x16,x17} | No EEG Answer sets |
| 122 | {x11,x14,x15,x16,x17} | No EEG Answer sets |

| | | |
|---|---|---|
| 123 | {x12,x14,x15,x16,x17} | No EEG Answer sets |
| 124 | {x11,x12,x14,x15,x16,x17} | {x11, x12, x14, x15, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, cefalea, -resfriado} |
| 125 | {x13,x14,x15,x16,x17} | No EEG Answer sets |
| 126 | {x11,x13,x14,x15,x16,x17} | No EEG Answer sets |
| 127 | {x12,x13,x14,x15,x16,x17} | No EEG Answer sets |
| 128 | {x11,x12,x13,x14,x15,x16,x17} | {x11, x12, x13, x14, x15, x16, x17, mialgia, dolor_lumbar, tos_seca, fiebre_aumento, -resfriado} |
| 129 | {x21} | No EEG Answer sets |
| 130 | {x11,x21} | No EEG Answer sets |
| 131 | {x12,x21} | No EEG Answer sets |
| 132 | {x11,x12,x21} | {x11, x12, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, odinofalgia, irritacion_ocular, -resfriado, gripe} |
| 133 | {x13,x21} | No EEG Answer sets |
| 134 | {x11,x13,x21} | No EEG Answer sets |
| 135 | {x12,x13,x21} | No EEG Answer sets |
| 136 | {x11,x12,x13,x21} | {x11, x12, x13, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, estornudos ,rinorrea, odinofalgia, irritacion_ocular, -resfriado} |
| 137 | {x14,x21} | No EEG Answer sets |
| 138 | {x11,x14,x21} | No EEG Answer sets |
| 139 | {x12,x14,x21} | No EEG Answer sets |
| 140 | {x11,x12,x14,x21} | {x11, x12, x14, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, rinorrea, odinofalgia, irritacion_ocular, -resfriado, gripe} |
| 141 | {x13,x14,x21} | No EEG Answer sets |
| 142 | {x11,x13,x14,x21} | No EEG Answer sets |
| 143 | {x12,x13,x14,x21} | No EEG Answer sets |
| 144 | {x11,x12,x13,x14,x21} | {x11, x12, x13, x14, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, rinorrea, odinofalgia, irritacion_ocular, -resfriado} |
| 145 | {x15,x21} | No EEG Answer sets |
| 146 | {x11,x15,x21} | No EEG Answer sets |
| 147 | {x12,x15,x21} | No EEG Answer sets |
| 148 | {x11,x12,x15,x21} | {x11, x12, x15, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, odinofalgia,irritacion_ocular, -resfriado} |
| 149 | {x13,x15,x21} | No EEG Answer sets |
| 150 | {x11,x13,x15,x21} | No EEG Answer sets |
| 151 | {x12,x13,x15,x21} | No EEG Answer sets |
| 152 | {x11,x12,x13,x15,x21} | {x11, x12, x13, x15, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, estornudos, odinofalgia, irritacion_ocular, -resfriado} |
| 153 | {x14,x15,x21} | No EEG Answer sets |
| 154 | {x11,x14,x15,x21} | No EEG Answer sets |
| 155 | {x12,x14,x15,x21} | No EEG Answer sets |

| | | |
|---|---|---|
| 156 | {x11,x12,x14,x15,x21} | {x11, x12, x14, x15, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, odinofalgia, irritacion_ocular, -resfriado} |
| 157 | {x13,x14,x15,x21} | No EEG Answer sets |
| 158 | {x11,x13,x14,x15,x21} | No EEG Answer sets |
| 159 | {x12,x13,x14,x15,x21} | No EEG Answer sets |
| 160 | {x11,x12,x13,x14,x15,x21} | {x11, x12, x13, x14, x15, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, odinofalgia, irritacion_ocular, -resfriado} |
| 161 | {x16,x21} | No EEG Answer sets |
| 162 | {x11,x16,x21} | No EEG Answer sets |
| 163 | {x12,x16,x21} | No EEG Answer sets |
| 164 | {x11,x12,x16,x21} | {x11, x12, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, irritacion_ocular, -resfriado} |
| 165 | {x13,x16,x21} | No EEG Answer sets |
| 166 | {x11,x13,x16,x21} | No EEG Answer sets |
| 167 | {x12,x13,x16,x21} | No EEG Answer sets |
| 168 | {x11,x12,x13,x16,x21} | {x11, x12, x13, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, estornudos, rinorrea, irritacion_ocular, -resfriado} |
| 169 | {x14,x16,x21} | No EEG Answer sets |
| 170 | {x11,x14,x16,x21} | No EEG Answer sets |
| 171 | {x12,x14,x16,x21} | No EEG Answer sets |
| 172 | {x11,x12,x14,x16,x21} | {x11, x12, x14, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, rinorrea, irritacion_ocular, -resfriado} |
| 173 | {x13,x14,x16,x21} | No EEG Answer sets |
| 174 | {x11,x13,x14,x16,x21} | No EEG Answer sets |
| 175 | {x12,x13,x14,x16,x21} | No EEG Answer sets |
| 176 | {x11,x12,x13x,14,x16,x21} | {x11, x12, x13, x14, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, rinorrea, irritacion_ocular, -resfriado} |
| 177 | {x15,x16,x21} | No EEG Answer sets |
| 178 | {x11,x15,x16,x21} | No EEG Answer sets |
| 179 | {x12,x15,x16,x21} | No EEG Answer sets |
| 180 | {x11,x12,x15,x16,x21} | {x11, x12, x15, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, irritacion_ocular, -resfriado} |
| 181 | {x13,x15,x16,x21} | No EEG Answer sets |
| 182 | {x11,x13,x15,x16,x21} | No EEG Answer sets |

| | | |
|---|---|---|
| 183 | {x12,x13,x15,x16,x21} | No EEG Answer sets |
| 184 | {x11,x12,x13,x15,x16,x21} | {x11, x12, x13, x15, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, estornudos, irritacion_ocular, -resfriado} |
| 185 | {x14,x15,x16,x21} | No EEG Answer sets |
| 186 | {x11,x14,x15,x16,x21} | No EEG Answer sets |
| 187 | {x12,x14,x15,x16,x21} | No EEG Answer sets |
| 188 | {x11,x12,x14,x15,x16,x21} | {x11, x12, x14, x15, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, irritacion_ocular, -resfriado} |
| 189 | {x13,x14,x15,x16,x21} | No EEG Answer sets |
| 190 | {x11,x13,x14,x15,x16,x21} | No EEG Answer sets |
| 191 | {x12,x13,x14,x15,x16,x21} | No EEG Answer sets |
| 192 | {x11,x12,x13,x14,x15,x16,x21} | {x11, x12, x13, x14, x15, x16, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, irritacion_ocular, -resfriado} |
| 193 | {x17,x21} | No EEG Answer sets |
| 194 | {x11,x17,x21} | No EEG Answer sets |
| 195 | {x12,x17,x21} | No EEG Answer sets |
| 196 | {x11,x12,x17,x21} | {x11, x12, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, odinofalgia, -resfriado} |
| 197 | {x13,x17,x21} | No EEG Answer sets |
| 198 | {x11,x13,x17,x21} | No EEG Answer sets |
| 198 | {x11,x13,x17,x21} | No EEG Answer sets |
| 199 | {x12,x13,x17,x21} | No EEG Answer sets |
| 200 | {x11,x12,x13,x17,x21} | {x11, x12, x13, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, estornudos, rinorrea, odinofalgia, -resfriado} |
| 201 | {x14,x17,x21} | No EEG Answer sets |
| 202 | {x11,x14,x17,x21} | No EEG Answer sets |
| 203 | {x12,x14,x17,x21} | No EEG Answer sets |
| 204 | {x11,x12,x14,x17,x21} | {x11, x12, x14, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, rinorrea, odinofalgia, -resfriado} |
| 205 | {x13,x14,x17,x21} | No EEG Answer sets |
| 206 | {x11,x13,x14,x17,x21} | No EEG Answer sets |
| 207 | {x12,x15,x17,x21} | No EEG Answer sets |
| 208 | {x11,x12,x15,x17,x21} | {x11, x12, x15, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, odinofalgia, -resfriado} |
| 209 | {x15,x17,x21} | No EEG Answer sets |

| | | |
|---|---|---|
| 210 | {x11,x15,x17,x21} | No EEG Answer sets |
| 211 | {x12,x15,x17,x21} | No EEG Answer sets |
| 212 | {x11,x12,x15,x17,x21} | {x11, x12, x15, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, odinofalgia, -resfriado} |
| 213 | {x13,x15,x17,x21} | No EEG Answer sets |
| 214 | {x11,x13,x15,x17,x21} | No EEG Answer sets |
| 215 | {x12,x14,x15,x17,x21} | No EEG Answer sets |
| 216 | {x11,x12,x13,x15,x17,x21} | {x11, x12, x13, x15, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento,estornudos, odinofalgia, -resfriado} |
| 217 | {x14,x15,x17,x21} | No EEG Answer sets |
| 218 | {x11,x14,x15,x17,x21} | No EEG Answer sets |
| 219 | {x12,x14,x15,x17,x21} | No EEG Answer sets |
| 220 | {x11,x12,x14,x15,x17,x21} | {x11, x12, x14, x15, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, odinofalgia, -resfriado} |
| 221 | {x13,x14,x15,x17,x21} | No EEG Answer sets |
| 222 | {x11,x13,x14,x15,x17,x21} | No EEG Answer sets |
| 223 | {x12,x13,x14,x15,x17,x21} | No EEG Answer sets |
| 224 | {x11,x12,x13,x14,x15,x17,x21} | {x11, x12, x13, x14, x15, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, odinofalgia, -resfriado} |
| 225 | {x16,x17,x21} | No EEG Answer sets |
| 226 | {x11,x16,x17,x21} | No EEG Answer sets |
| 227 | {x12,x16,x17,x21} | No EEG Answer sets |
| 228 | {x11,x12,x16,x17,x21} | {x11, x12, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, rinorrea, -resfriado} |
| 229 | {x13,x16,x17,x21} | No EEG Answer sets |
| 230 | {x11,x13,x16,x17,x21} | No EEG Answer sets |
| 231 | {x12,x13,x16,x17,x21} | No EEG Answer sets |
| 232 | {x11,x12,x13,x16,x17,x21} | {x11, x12, x13, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento,estornudos, rinorrea, -resfriado} |
| 233 | {x14,x16,x17,x21} | No EEG Answer sets |
| 234 | {x11,x14,x16,x17,x21} | No EEG Answer sets |
| 235 | {x12,x14,x16,x17,x21} | No EEG Answer sets |
| 236 | {x11,x12,x14,x16,x17,x21} | {x11, x12, x14, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, rinorrea, -resfriado} |

| | | |
|---|---|---|
| 237 | {x13,x14,x16,x17,x21} | No EEG Answer sets |
| 238 | {x11,x13,x14,x16,x17,x21} | No EEG Answer sets |
| 239 | {x12,x13,x14,x16,x17,x21} | No EEG Answer sets |
| 240 | {x11,x12,x13,x14,x16,x17,x21} | {x11, x12, x13, x14, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, rinorrea, -resfriado} |
| 241 | {x15,x16,x17,x21} | No EEG Answer sets |
| 242 | {x11,x15,x16,x17,x21} | No EEG Answer sets |
| 243 | {x12,x15,x16,x17,x21} | No EEG Answer sets |
| 244 | {x11,x12,x15,x16,x17,x21} | {x11, x12, x15, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, estornudos, -resfriado} |
| 245 | {x13,x15,x16,x17,x21} | No EEG Answer sets |
| 246 | {x11,x13,x15,x16,x17,x21} | No EEG Answer sets |
| 247 | {x12,x13,x15,x16,x17,x21} | No EEG Answer sets |
| 248 | {x11,x12,x13,x15,x16,x17,x21} | {x11, x12, x13, x15, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, estornudos, -resfriado} |
| 249 | {x14,x15,x16,x17,x21} | No EEG Answer sets |
| 250 | {x11,x14,x15,x16,x17,21} | No EEG Answer sets |
| 251 | {x12,x14,x15,x16,x17,x21} | No EEG Answer sets |
| 252 | {x11,x12,x14,x15,x16,x17,x21} | {x11, x12, x14, x15, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, cefalea, -resfriado} |
| 253 | {x13,x14,x15,x16,x17,x21} | No EEG Answer sets |
| 254 | {x11,x13,x14,x15,x16,x17,x21} | No EEG Answer sets |
| 255 | {x12,x13,x14,x15,x16,x17,x21} | No EEG Answer sets |
| 256 | {x11,x12,x13,x14,x15,x16,x17,x21} | {x11, x12, x13, x14, x15, x16, x17, mialgia, dolor_lumbar, x21, tos_seca, fiebre_aumento, -resfriado} |

# Bibliografía


[1] Christopher Cherniak. Minimal rationality. 1986.

[2] Matthew L. Ginsberg. Counterfactuals. Artificial intelligent. 1995.

[3] Esra Erdem y Vladimir Lifschitz. Fages theorem for programs with nested expressions. *In Proceedings of the 17th International Conference on Logic Programming*, Springer:242–254, December 2001.

[4] Dr. Alejandro Ramírez F. La abducción y el conocimiento tácito.

[5] Stan Franklin y Art Graesser. Is it an agent, or just a program? a taxonomy for autonomus agents. proceedings of the third international workshop on agent theories, architectures, and languages. 1996.

[6] M. Gelfond y V. Lifschitz. The stable model semantics for logic programming. *In R. Kowalski and K. Bowen*, MIT Press, 1988.

[7] Michael Gelfond y Vladimir Lifschitz. The stable models semantics for logic programming. *5th Conference on Logic Programming*, págs. 1070–1080, 1988.

[8] Seymour Lipschutz. Teora de conjuntos y temas afines. *McGRAW-HILL/Interamericana*, 1991.

[9] M. Osorio y V. Cuevas. Updates in answer set programming: An approach based on basic structural properties. pág. 7(04):451479, July 2007.

[10] Mauricio Osorio y Claudia Zepeda. Minimal extended generalized answer sets for update sequences.

[11] Fernando G. Torres. Las distintas miradas acerca de la racionalidad instrumental mínima y la ética. 2006.

[12] Fernando Soler Toscano. Razonamiento abductivo en lógica clasica.

[13] Francisco Javier Tobon Vazquez. Pizarron electrónico usando agentes inteligentes. 2006.

[14] L. R. Tang Vladimir Lifschitz y H. Turner. Annals of mathematics and articial intelligence. *Nested expressions in logic programs*, págs. 369–389, 1999.

[15] M. Wooldridge y N. Jennings. Intelligent agents: Theory and practice. 1995.