Aplicación de Operaciones de Cambio en Sistemas Basados en Conocimiento

Martín O. Moguillansky Marcelo A. Falappa*

Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Av. Alem 1253 - (B8000CPB) Bahía Blanca - Argentina
Phone/Fax: (+54)(291)459-5136
E-mail: martillansky@hotmail.com mfalappa@cs.uns.edu.ar

Resumen

La teoría del cambio de creencias modela la dinámica del conocimiento, es decir, el estado de las creencias de un agente luego de recibir información externa. Uno de los principales modelos de cambio que existen en la teoría de cambio de creencias es el modelo AGM, donde la nueva información tiene prioridad sobre la existente, aunque existen diversidad de modelos alternativos que tratan de remediar esto.

El presente trabajo trata acerca de la implementación del modelo de cambio utilizado en operaciones de cambio priorizadas y no-priorizadas. Dicho modelo toma como referencia el modelo AGM y otras operaciones no clásicas, además de conceptos de diferentes autores para definir los mecanismos de inferencia utilizados en la implementación.

1. El Modelo de Cambio

Conocemos como **bases de creencias** a aquellos estados de conocimiento representados mediante un conjunto de sentencias no necesariamente clausurado. Asimismo, sabemos que los **conjuntos de creencias** son un conjunto de sentencias de un lenguaje determinado clausuradas bajo cierta operación de consecuencia lógica. En general, los conjuntos de creencias son infinitos y por tal motivo, es imposible tratar con ellos en una computadora. En su lugar, es posible caracterizar las propiedades que debe satisfacer cada una de las operaciones de cambio sobre representaciones finitas de un estado de conocimiento.

Las operaciones de **expansión** son las más sencillas de definir desde un punto de vista lógico. Consiste en agregar una nueva creencia al estado epistémico, sin garantizar la consistencia del mismo luego de su aplicación. El símbolo "+" generalmente es usado para representar a un operador de expansión.

*Miembro del LIDIA (Laboratorio de Investigación y Desarrollo en Inteligencia Artificial) y del IICyTI (Instituto de Investigación en Ciencia y Tecnología Informática).

Las operaciones de **contracción** eliminan del estado epistémico una creencia α y aquellas creencias que posibilitan su deducción o inferencia. Es de particular importancia que las sentencias a eliminar representen el mínimo cambio sobre el estado epistémico. El símbolo "$-$" generalmente es usado para representar a un operador de contracción.

Las operaciones de **revisión** consisten en la inserción de sentencias al estado epistémico con la garantía de que si el mismo es consistente antes de la revisión, la consistencia se mantendrá luego de ella. Esto significa que una revisión agrega una nueva creencia y tal vez elimine otras para evitar inconsistencias.

A continuación, mencionaremos los postulados básicos para revisiones formulados en el modelo AGM [AGM85]. En estos postulados se asume que el estado de conocimiento de un agente se representa mediante un conjuntos de creencias $\mathbf{K}$ (esto es, $\mathbf{K} = Cn(\mathbf{K})$), "$*$" representa un operador de revisión, y α, β sentencias del lenguaje.

(K^*1) **Clausura:** Para cualquier conjunto de creencias $\mathbf{K}$ y cualquier sentencia α, $\mathbf{K} * \alpha$ es un conjunto de creencias.

(K^*2) **Exito:** $\alpha \in \mathbf{K} * \alpha$.

(K^*3) **Inclusión:** $\mathbf{K} * \alpha \subseteq \mathbf{K} + \alpha$.

(K^*4) **Vacuidad:** Si $\mathbf{K} \nvdash \neg\alpha$ entonces $\mathbf{K} * \alpha = \mathbf{K} + \alpha$.

(K^*5) **Consistencia:** Si $\nvdash \neg\alpha$ entonces $\mathbf{K} * \alpha \neq \mathbf{K}_\perp$.

(K^*6) **Extensionalidad/Preservación:** Si $Cn(\alpha) = Cn(\beta)$ entonces $\mathbf{K} * \alpha = \mathbf{K} * \beta$.

La idea de definir una operación de cambio a partir de otra se fundamenta en la búsqueda de la simplificación y coherencia de la teoría epistemológica. A continuación, veremos que las contracciones y revisiones son interdefinibles entre sí, asumiendo a las expansiones como operaciones de cambio primitivas:

- **Identidad de Levi:** $K * \alpha = (K - \neg\alpha) + \alpha$.
- **Identidad de Harper:** $K - \alpha = K \cap K * \neg\alpha$.

Estas identidades indican que la definición de un método algorítmico para computar uno de los procesos puede derivar en la construcción del otro.

2. Kernel Contractions

El operador de Kernel Contraction es aplicable tanto a bases de creencias como a conjuntos de creencias. Consiste en construir un operador de contracción que sea capaz de seleccionar y eliminar entre los elementos de K creencias que contribuyen a la inferencia de α.

Definición 2.1 [Han93]: Sean K un conjunto de sentencias y α una sentencia. El conjunto $K^\perp\alpha$, denominado *conjunto de kernels* es el conjunto de conjuntos K' tales que:

1. $K' \subseteq K$.

2. $K' \vdash \alpha$.

3. Si $K'' \subset K'$ entonces $K'' \nvdash \alpha$.

El conjunto $K^{\perp}\alpha$ también se denomina *conjunto de α-kernels* y cada uno de sus elementos se denomina *α-kernel*.

Para que una operación de contracción sea exitosa se necesita eliminar al menos un elemento de cada α-kernel. Los elementos a ser eliminados son seleccionados por una **función de incisión**, denominada así porque realiza un corte sobre cada kernel.

Definición 2.2 [Han93]: Una función "σ" es una *función de incisión* para un conjunto K si para toda sentencia α vale que:

1. $\sigma(K^{\perp}\alpha) \subseteq \bigcup(K^{\perp}\alpha)$.

2. Si $K' \in K^{\perp}\alpha$ y $K' \neq \emptyset$ entonces $K' \cap \sigma(K^{\perp}\alpha) \neq \emptyset$.

Una vez que la función de incisión fue aplicada, debemos eliminar de la base K aquellas sentencias que la función de incisión indique. Esto es, luego de una contracción de K por α deberían conservarse todas aquellas sentencias de K que no fueron seleccionadas por σ.

Definición 2.3 [Han93]: Sea K un conjunto de sentencias, α una sentencia cualquiera, y $K^{\perp}\alpha$ el conjunto de α-kernels de K. Sea "σ" una función de incisión para K. El operador "$-_{\sigma}$", denominado *kernel contraction determinado por* "σ" se define de la siguiente manera:

(**Def** Kernel Contraction) $K -_{\sigma} \alpha = K \backslash \sigma(K^{\perp}\alpha)$.

Un operador "$-$" es un operador de kernel contraction para K si y sólo si existe alguna función de incisión "σ" tal que $K - \alpha = K -_{\sigma} \alpha$ para toda sentencia α.

2.1. Funciones de Incisión Cualitativas y Cuantitativas

Una función de incisión σ selecciona un elemento de cada subconjunto perteneciente a $K^{\perp}\alpha$, para evitar la deducción de α. Con tal fin, podemos contar con diferentes políticas de selección [FS02]:

- **Cualitativas:** se descartarán primero aquellas sentencias de menor *importancia epistémica.*
- **Cuantitativas:** se pretende minimizar el número de sentencias a ser eliminadas. Para ello, se seleccionarán las sentencias que se encuentren repetidas más cantidad de veces en distintos α-kernels.

Definición 2.1.1 : Sea K un conjunto de sentencias y σ una función de incisión para K. Decimos que σ es una función de incisión cuantitativamente óptima si, para cualquier otra función de incisión σ' de K, y para toda sentencia $\alpha \in \mathcal{L}$ se verifica que:

$$|\sigma(K^{\perp}\alpha)| \leq |\sigma'(K^{\perp}\alpha)|$$

Esto es, la cantidad de elementos eliminados por σ es la mínima.

2.2. Importancia Epistémica

Este método consiste en proveer una forma de comparar creencias a partir de la importancia de permanencia de la misma en el estado epistémico, de manera que al realizar una contracción con respecto a determinada creencia, se eliminen aquellas sentencias con más baja importancia epistémica.

Para poder definir un operador de importancia epistémica, asumiremos que los distintos grados de importancia epistémica estarán medidos cualitativamente y no cuantitativamente. Por ejemplo, si α y β son sentencias en $\mathfrak{L}$, la relación estricta $\alpha < \beta$ expresará que *"β es más importante epistémicamente que α"*.

2.2.1. Condicionales vs. Hechos

Cuando hablamos de ***contenido intensional*** de una sentencia condicional del tipo $A \rightarrow B$, donde tanto A como B contienen variables, es necesario suponer que tal regla afectará a un subconjunto de elementos del conjunto de creencias al que se refiera. Es decir, que tales sentencias se refieren a todo un universo de objetos.

En cambio, nos referimos a una sentencia disyuntiva de fórmulas atómicas desde su ***contenido extensional***, dado que en ese caso una sentencia no se refiere mas allá de los literales que en ella estén expresados. Es decir, tales sentencias se refieren a ciertos objetos en particular de todo un universo de objetos.

Dado que las sentencias condicionales (*contenido intensional*) son más generales que las sentencias disyuntivas o hechos (*contenido extensional*), nuestro sistema determinará en forma implícita un orden de importancia epistémica entre condicionales y disyunciones o hechos, de manera que siempre será más importante epistémicamente una sentencia condicional, a menos que el usuario determine lo contrario explícitamente.

3. Incorporación Consistente de Creencias

Definiremos la operación de revisión en un conjunto K con respecto a una sentencia α, mediante la *Identidad de Levi*, asumiendo que " $-$ " es un operador de *kernel contraction* determinado por una función de incisión "σ".

Definición 3.1 [Han96]: Sea " $-$ " una kernel contraction para un conjunto K. Entonces el operador de *kernel revision interno* para K se define como sigue:

$$K \mp_\sigma \alpha = (K - \neg\alpha) + \alpha.$$

La notación " $\mp$ ", indica que el operador de revisión primero realiza una contracción y luego una expansión.

La implementación del sistema está basada en la aplicación de las operaciones de revisión mediante el operador de *kernel revision interno*. Para definir formalmente tal operador necesitaremos el uso de dos postulados adicionales propuestos por Hansson:

- **Retención de Núcleo [Han93]:** Si $\beta \in K$ y $\beta \notin K * \alpha$ entonces existe algún K' tal que $K' \subseteq (K \cup \{\alpha\})$, K' es consistente pero $K' \cup \{\alpha\}$ es inconsistente.

- **Uniformidad [Han93]:** Si para todo $K' \subseteq K$, $K' \cup \{\alpha\}$ es inconsistente si y sólo si $K' \cup \{\beta\}$ es inconsistente, entonces $K \cap (K * \alpha) = K \cap (K * \beta)$.

El siguiente teorema da una caracterización axiomática para el operador de *kernel revision interno* aplicado sobre conjuntos no necesariamente clausurados.

Teorema 3.1 [Han93]: El operador " $*$ " es un operador de *kernel revision interno* para una base de creencias K si y sólo si satisface *inclusión*, *retención de núcleo*, *éxito* y *uniformidad*.

4. Revisiones No Priorizadas

Las operaciones de revisión clásicas están caracterizadas por ciertos postulados de racionalidad introducidos por Gärdenfors [Gär88], algunos de los cuales, han sido cuestionados por considerarse arbitrarios. En particular, el postulado de *éxito* establece que la nueva información a ser revisada en un estado epistémico deberá formar parte del mismo, a pesar de que para mantener la consistencia de la misma se deban dejar de lado otras creencias que ya pertenecen al estado de conocimiento del agente.

Por tal motivo, es interesante la definición de nuevos tipos de operaciones de revisión que logren capturar la información de manera más intuitiva que el operador de revisión clásico, de manera que una nueva información no tenga absoluta prioridad sobre la contenida en el estado epistémico. Para establecer un ejemplo concreto, pensemos en un ser racional. Dicho ser constantemente está en proceso de aprendizaje, captando todo nuevo conocimiento de forma incondicional. Pero ¿qué sucede cuando una nueva creencia contradice conocimientos ya adoptados en él? En muchos casos, se exigiría una explicación para la nueva creencia, a partir de la cual poder tomar una decisión con respecto a la contradicción ocurrida, de modo de renovar sus creencias en forma consistente y darle más sustento a la nueva creencia.

Antes de presentar el nuevo operador de revisión, presentaremos la noción de explicación utilizada en [FKS02]. En toda explicación existen dos elementos distinguibles: el **explanans**, encargado de representar la explicación propiamente dicha, y el **explanandum**, que representa la conclusión final de la explicación.

Definición 4.1 [FKS02]: El conjunto A es una explicación para α si y sólo si A verifica:

1. **Deducción:** $A \vdash \alpha$. Es la esencia de la explicación por si misma.
2. **Consistencia:** $A \nvdash \bot$. Es necesaria en el explanans para impedir que un conjunto inconsistente sea explicación para cualquier sentencia.
3. **Minimalidad:** Si $B \subset A$ entonces $B \nvdash \alpha$. Se exigirá en el explanans, para evitar información irrelevante para el explanandum.
4. **Contenido Informacional:** $\alpha \nvdash A$. Evita que una sentencia se explique a si misma o que, se den casos en que A sea, por ejemplo, $\{\alpha \vee \beta, \alpha \vee \neg\beta\}$.

La relación *"A es una explicación para α"* será notada como $A \rightarrowtail \alpha$.

En diálogos entre agentes es común que un agente no acepte totalmente lo que le informa otro sino que acepte parte de esa información (*aceptación parcial*). Por tal motivo, utilizaremos un operador de revisión múltiple que permita modelar este comportamiento.

Sea A una explicación para α. Sean K y A conjuntos de sentencias. Sea "$\diamond$" un operador de revisión mediante explicaciones. Los siguientes postulados han sido propuestos para este tipo de operador [FKS02]:

1. **Inclusión:** $K \diamond A \subseteq K \cup A$.
2. **Retención de Núcleo:** Si $\alpha \in (K \cup A) \backslash (K \diamond A)$ entonces existe un conjunto H tal que $H \subseteq (K \cup A)$, H es consistente y $H \cup \{\alpha\}$ es inconsistente.
3. **Congruencia:** Si $K \cup A = K \cup B$ entonces $K \diamond A = K \diamond B$.
4. **Imparcialidad:** Si $A \nvdash \perp$ y $B \nvdash \perp$ y para todo $H \subseteq K$ vale que $(H \cup A) \vdash \perp$ si y sólo si $(H \cup B) \vdash \perp$ entonces $(H \cup A) \backslash (K \diamond A) = (H \cup B) \backslash (K \diamond B)$.
5. **Reversión:** Si $K \cup A$ y $K \cup B$ tienen los mismos conjuntos minimalmente inconsistentes entonces $(H \cup A) \backslash (K \diamond A) = (H \cup B) \backslash (K \diamond B)$.
6. **Preservación de Consistencia:** Si $K \nvdash \perp$ entonces $K \diamond A \nvdash \perp$.

A continuación, veremos un algoritmo que expresará un posible desarrollo del operador de revisión mediante explicaciones con aceptación parcial:

1. Inicialmente se acepta el conjunto de entrada A.
2. Se eliminan de $K \cup A$ todas las posibles contradicciones.

Siguiendo el algoritmo, si K es consistente, al insertar A en ella obtendremos posiblemente un estado intermedio inconsistente. Luego, se restaurará la consistencia en el estado epistémico aplicando la operación de *consolidación*[1].

4.1. Kernel Revision Mediante Explicaciones

Para poder definir el operador de kernel revision mediante explicaciones necesitamos contar con una función de incisión aplicable a cualquier conjunto del lenguaje. Por lo tanto definiremos una función de incisión global (esto es, una función de incisión que pueda aplicarse a cualquier subconjunto del lenguaje).

Definición 4.1.1 [FKS02]: Una función de incisión global es una función "σ" tal que para todo conjunto $H \subseteq \mathfrak{L}$ y toda sentencia $\alpha \in \mathfrak{L}$ vale que:

1. $\sigma(H^{\perp}\alpha) \subseteq \bigcup(H^{\perp}\alpha)$.
2. Si $X \in H^{\perp}\alpha$ y $X \neq \emptyset$ entonces $(X \cap \sigma(H^{\perp}\alpha)) \neq \emptyset$.

[1] Adaptaremos en forma directa las definiciones de Kernel Contraction para definir una operación de *Kernel Consolidation*. En forma intuitiva, una operación de kernel consolidation consiste en encontrar todos los subconjuntos minimales de K que derivan $\perp$ y cortar cada uno de ellos mediante una función de incisión.

Definida ya la función de incisión global, estamos en condiciones de establecer una definición formal para el operador de kernel revision mediante explicaciones.

Definición 4.1.2 [FKS02]: Sean K y A conjuntos de sentencias y "σ" una función de incisión global. El operador "$\diamond$" de kernel revision mediante explicaciones ($\diamond : 2^{\mathcal{L}} \times 2^{\mathcal{L}} \Rightarrow 2^{\mathcal{L}}$) se define como:

(**Def** Kernel Revision mediante Explicaciones) $K \diamond A = (K \cup A) \backslash \sigma((K \cup A)^{\perp} \perp)$

Habiendo presentado las definiciones formales necesarias para las operaciones de *kernel revision mediante explicaciones*, sólo resta dar una caracterización axiomática para ellas. El siguiente teorema da la representación correspondiente:

Teorema 4.1.1 [FKS02]: Sean K un conjunto de sentencias consistente. El operador "$\diamond$" es un operador de kernel revision mediante explicaciones si y sólo si satisface *inclusión, consistencia, retención de núcleo* y *reversión.*

Figura 1: Operación de Revisión No Priorizada.

5. Representación del Conocimiento

El sistema representará el conocimiento de un agente mediante una **base de creencias**, llevando registro de las sentencias deducidas mediante la clausura lógica en forma discriminada de la base misma. En el conjunto de **creencias deducidas** sólo se encontrarán aquellas sentencias deducidas a partir de la ejecución de los mecanismos de inferencia aplicados sobre la base de creencias y sobre el mismo conjunto de creencias deducidas. Sólo serán parte de este conjunto aquellas disyunciones deducidas por los mecanismos de inferencia que son un paso intermedio en la deducción de hechos, quienes serán mostrados como parte de la clausura. Mediante esta restricción, se evitará el problema de inviabilidad computacional para la clausura lógica.

El lenguaje $\mathfrak{L}$ está compuesto por un subconjunto del lenguaje del *cálculo de predicados*. Se utilizarán *fórmulas bien formadas* (fbfs) para expresar las creencias del conocimiento. Tales sentencias serán formadas a partir de los **conectivos lógicos** " `y` ", " `o` ", " `no` ", " `->` ", y también a partir de **identificadores de proposiciones y predicados**.

Además, dos creencias estarán relacionadas epistémicamente mediante pares $(u, v) \in R$, donde R puede ser uno de los **conjuntos relacionales** " $<$ ", " $>$ " ó " $=$ ".

6. Mecanismos de Inferencia

A continuación, presentaremos los mecanismos de inferencia utilizados en el sistema. Es necesario que el lector esté familiarizado con conceptos relacionados a programación en lógica, cálculo de predicados, y procesos computacionales en los mismos. La notación será similar a PROLOG, los predicados y las constantes comenzarán con letras minúsculas, mientras que las variables comenzarán con letras mayúsculas. Las reglas de cuantificación de las variables también son equivalentes a las utilizadas en PROLOG.

Definiremos un procedimiento de inferencia basado en *resolución de literales complementarios*. Si existen dos sentencias de la forma α `o` β, y δ `o` ϵ y existe una sustitución de variables φ tal que $\beta =$ (`no` $\delta)[\varphi]$ [2] entonces se obtiene una nueva resolvente $(\alpha$ `o` $\epsilon)[\varphi]$. Notemos que este tipo de sentencias disyuntivas de literales no básicos pueden obtenerse transformando las reglas de la forma A `->` B en disyunciones de la forma `no` A `o` B.

El procedimiento de resolución toma como argumento a un conjunto de expresiones en una versión simplificada del cálculo de predicados, llamada *forma clausal*. Una cláusula es un conjunto de literales representando sus disyunciones. Por ejemplo, los conjuntos $\{pajaro(toto)\}$ y $\{$ `no` $pajaro(X), vuela(X)\}$ son las cláusulas que corresponden a las sentencias $pajaro(toto)$ y $pajaro(X)$ `->` $vuela(X)$ respectivamente.

Dentro de los métodos de resolución conocidos, podemos utilizar el método de *resolución ordenada* [GN88]. El mismo consiste en una estrategia de resolución muy restrictiva en la cual cada cláusula es tratada como un conjunto linealmente ordenado. La resolución está permitida sólo en el primer literal de cada cláusula. Los literales en una nueva conclusión son ordenados mediante el mismo criterio. Luego se aplica la estrategia en forma recursiva hasta que no se obtengan nuevas conclusiones.

Debido a la forma ordenada de las cláusulas que intervienen en una deducción y el hecho de observar sólo el primer literal de cada una para obtener una nueva cláusula hace de la *resolución ordenada* una estrategia extremadamente eficiente.

Sin embargo, este método puede resultar no completo con cláusulas no Horn. En caso de aplicar resolventes con cláusulas no Horn, el sistema adoptará el método de *resolución por conjunto soporte* [GN88].

7. Mecanismos de Orden

Denominamos mecanismos de orden a aquellos conceptos por los que se efectivizan las relaciones binarias entre creencias, pertenecientes a la base de conocimiento, que representan la importancia epistémica del sistema.

[2] En general, la notación $\alpha[\varphi]$ indica que se aplica la sustitución φ a la fórmula α.

Las siguientes funciones serán necesarias para lograr una asociación entre las diferentes relaciones binarias definidas en el sistema.

Definición 7.1 [Mom03]: Dada la función $\xi : K \times K \Rightarrow K \times K$, llamada *función de inversión de orden*, llamaremos al conjunto $\xi(R) = \{(w, v) : (v, w) \in R\}$, *conjunto de orden invertido sobre la relación R.*

A partir de la definición anterior podemos unificar las relaciones "<" y ">" mediante la identidad $<=< \cup \xi(>)$. De esta manera, obtenemos una forma de conversión entre las relaciones "<" y ">", de manera que si tuviéramos las relaciones $a < b$ y $b > c$, mediante el uso de la identidad anterior obtendríamos las relaciones $a < b$ y $c < b$, resumiendo así el uso de ambos operadores relacionales a sólo uno de ellos.

Ahora debemos dar una definición para lograr la asociación de la relación binaria restante "=" con las relaciones "<" y ">", de manera que logremos obtener un *conjunto de relaciones únicas* resultante de la asociación de todos los pares pertenecientes a toda relación binaria del sistema. τ^R será la relación que se obtiene de insertar las relaciones generadas al reemplazar cada aparición de una creencia en una relación R por su igual en el conjunto relacional "=". Por ejemplo, si se tuviesen las relaciones $a = b$, $a < c$ y $d < b$, el nuevo conjunto de relaciones resultante sería $a < c$, $d < b$, $b < c$ y $d < a$.

Un *conjunto de relaciones únicas* será la unificación de todas las relaciones binarias participantes del sistema de orden epistémico en la implementación, en un único tipo de relación (en la implementación esa relación es "<").

Definición 7.2 [Mom03]: Sea $\mathcal{R}$ el *conjunto de relaciones únicas* aplicadas a K, donde:

$$(\textbf{Def}\ \mathcal{R})\ \mathcal{R}\ =\ \tau(< \cup\ \xi(>))$$

Existe una asociación entre un grafo dirigido (digrafo) y una relación binaria, de manera que sea posible una examinación de las relaciones entre elementos y, más importante aún, brindar una forma alternativa para la comprobación computacional de la coherencia entre el orden de las creencias, fijado por las relaciones binarias ligadas a ellas.

Definición 7.3 [Mom03]: Sea $\vec{G}(R)$ el digrafo asociado a toda relación R tal que:

- $V = K$, el conjunto V de vértices de $\vec{G}$ es el estado epistémico K.
- Si $(v, w) \in R$ entonces hay un arco con origen v y extremo w en $\vec{G}$.

El sistema soportará internamente el orden indicado por las relaciones binarias contenidas en el conjunto de importancia epistémica, trabajando directamente sobre un **digrafo acíclico** $\vec{G}(\mathcal{R})$.

Es importante destacar que el sistema operará sobre este tipo de digrafos en forma correcta hasta el momento en que se encuentra con un ciclo en él. De ocurrir esto, el sistema habrá descubierto una *contradicción* entre las relaciones que conforman la importancia epistémica del sistema, informando el error en ese momento.

Estos errores se verifican antes del ingreso de una nueva relación entre dos creencias, es decir que ocurre cuando el conjunto de conocimientos está a punto de ser modificado.

Figura 2: Ejemplo de la visualización de las ventanas.

8. Representación Interna de los Conocimientos

Vistos los fundamentos teóricos que motivaron el desarrollo de nuestro trabajo, resta aclarar que su implementación lógica se realizó en PROLOG mientras que la interface gráfica y de comunicación con el usuario se programó en JAVA.

De esta manera, el usuario, al ejecutar el sistema, está corriendo un proceso con ejecución sobre la máquina virtual creada por JAVA y aquel, mediante los requerimientos del usuario, entablará la comunicación con PROLOG para el manejo de la base de creencias y la aplicación de las operaciones de cambio sobre ella.

Como ya hemos visto, los conocimientos en el sistema son manejados mediante dos conjuntos, la base de creencias y los hechos inferidos. Pero, ¿cómo están representados internamente? Las sentencias pertenecientes a la base de creencias estarán presentes en el sistema, como parámetros de un predicado dinámico llamado `kb`. Mientras que aquellas sentencias que formen parte de los hechos inferidos, ya sea en los estados intermedios o en el estado final, el cuál es de disposición del usuario, estarán como parámetro de un predicado dinámico llamado `fc`.

Ambos predicados dinámicos contienen un primer parámetro dedicado a la numeración de los conocimientos. Un secuenciamiento en las sentencias representantes de los conocimientos, permitió llevar un orden de examinación de los mismos y evitar repetir comparaciones entre dos sentencias ya examinadas.

Además, el predicado dinámico `kb` cuenta con un tercer parámetro, el cual es el encargado de llevar consigo las relaciones binarias que asocian a la misma sentencia con otras pertenecientes a la base de creencias. Esta labor se lleva a cabo mediante un conjunto (implementado con una lista), que contiene cada una de las relaciones en cuestión.

Por su parte, el tercer parámetro del predicado dinámico `fc` lleva cuenta de los números de creencias utilizadas para su propia deducción. De esta manera, este último parámetro también será utilizado en el caso de las consultas, mostrando las sentencias que sustentan la prueba de determinada creencia.

Por lo tanto definimos a los predicados dinámicos del sistema como sigue:

- La terna `kb`(N,S,IE) compuesta por:
  1. N, número de sentencia.
  2. S, sentencia que representa una creencia.
  3. IE, conjunto de relaciones de S con otras creencias.
- La terna `fc`(I,C,P) compuesta por:
  1. I, número de forma clausal.
  2. C, forma clausal deducida.
  3. P, conjunto de identificaciones de creencias que deducen C.

Entre las virtudes del sistema, podemos mencionar su versatilidad en la interacción con el usuario, permitiendo que el mismo sistema pueda realizar los cambios de manera automática (basándose en las relaciones de importancia epistémica almacenadas o en criterios cuantitativos) o de manera manual. En caso de cambios manuales, el sistema deja en manos del usuario la decisión de que creencias conservar o eliminar. No obstante, dependiendo de los criterios cualitativos o cuantitativos adoptados (establecidos en la configuración), el sistema indica las creencias que deberían ser eliminadas, siguiendo los criterios anteriormente establecidos. En esos casos, el usuario tendrá la decisión final sobre qué eliminar y qué mantener, pero será sustentado con información provista por el sistema.

9. Conclusiones

Durante el diseño del sistema se buscó la forma de evitar los problemas computacionales que están involucrados con el uso de conjuntos de creencias. Por este motivo, se utilizaron conjuntos de sentencias no clausurados, aunque se podían obtener clausuras discriminadas (restringidas) de la misma.

Para lograr una operación de clausura eficiente, se investigaron distintos mecanismos de inferencia. Entre ellos las reglas *Modus Ponens*, *Modus Tolens* y *Resolución de Literales Complementarios*, siendo esta última la única que finalmente fuera utilizada, dado que sus deducciones contienen a las inferidas por las dos primeras.

Con el fin de evitar las deducciones infinitas en la clausura, se decidió que el conjunto de clausura debería contener hechos solamente, evitando las disyunciones intermedias en una secuencia de sentencias inferidas. Pero ¿cómo decidir si una deducción concluiría,en algún momento de la prueba con un hecho, o si simplemente nos conduciría a una secuencia de prueba infinita? Por ello se trabajó sobre la prueba de sentencias pertenecientes al *Universo de Herbrand* asociado al conjunto de creencias. Si bien mediante este uso se logró dar con conjuntos de clausura finitos (por no contar con funciones en el lenguaje), la aplicación de la operación

en cuestión constituía una carga extra considerable para el tiempo de ejecución del sistema. Luego, se concluyó que la *resolución ordenada* y *resolución mediante conjunto soporte* serían apropiadas para la implementación del sistema, debido a detalles de la unificación de PROLOG y a las formas de manejo de las estructuras elegidas.

Finalmente, el sistema fue testeado con cientos de sentencias, para verificar que el uso de los mecanismos seleccionados en la implementación fueran los más apropiados, desde un punto de vista de eficiencia y correctitud.

Como trabajo futuro, desarrollaremos una aplicación servidora en internet que aplique los operadores de cambio anteriormente presentados esperando así, lograr un testeo exhaustivo del sistema, así como nuevas conclusiones y propuestas de desarrollo futuras.

Referencias

[AGM85] Carlos Alchourrón, Peter Gärdenfors, David Makinson. *On the logic of theory change: Partial Meet Contraction and Revision Functions.* The Journal of Symbolic Logic, 50: 510–530, 1985.

[FS02] Marcelo A. Falappa, Guillermo R. Simari. *Contracciones Kernel: Funciones de Incisión Cuantitativas.* V Workshop de Investigadores en Ciencias de la Computación, WICC'2003, Universidad Nacional del Centro, Mayo de 2003: 232-236.

[FKS02] Marcelo A. Falappa, Gabriele Kern-Isberner, Guillermo R. Simari. *Explanations, Belief Revision and Defeasible Reasoning.* Artificial Intelligence Journal, 141: 1–28, 2002.

[Gär88] Peter Gärdenfors. *Knowledge in Flux: Modelling the Dynamics of Epistemic States.* Cambridge, MA: The MIT Press, Bradford Books, 1988.

[Maf99] Marcelo A. Falappa. *Teoría de Cambio de Creencias y sus Aplicaciones sobre Estados de Conocimiento.* Tesis de Doctorado, Universidad Nacional del Sur, 1999.

[Maf92] Marcelo A. Falappa. *Una Implementación Computacional de Razonamiento No-Monotónico mediante la Revisión de Creencias.* SIAR'93: 10–19, 1993.

[Han97] Sven Ove Hansson. *Semi-Revision.* Journal of Applied Non-Classical Logic, 7: 151–175, 1997.

[Han96] Sven Ove Hansson. *A Textbook of Belief Dynamics: Theory Change and Database Updating.* 1998.

[Han93] Sven Ove Hansson. *Kernel Contraction.* The Journal of Symbolic Logic, 59: 845–859, 1994.

[GN88] Michael R. Genesereth, Nils J. Nilsson. *Logical Foundations of Artificial Intelligence. Stanford University.*, 1988.

[Mom03] Martín O. Moguillansky. *Implementación de Operaciones de Cambio Priorizadas y No-Priorizadas en Entornos Computacionales.* Tesis de Licenciatura, Universidad Nacional del Sur, 2003.