NO MONOTONÍA Y RAZONAMIENTO PLAUSIBLE, DOS ENFOQUES DE CARÁCTER CUALITATIVO PARA REALIZAR RAZONAMIENTO APROXIMADO

José Antonio Gámez Martín
José Miguel Puerta Callejón
Francisco José Vigo Bustos

José Antonio Gámez Martín, José Miguel Puerta Callejón y Francisco José Vigo Bustos están en el Dpto. de Informática, E. U. Politécnica de Albacete.

RESUMEN

Gran parte del conocimiento con el que los humanos razonan es inexacto o incierto en algún aspecto. Esta incertidumbre puede deberse a varios factores: a que el conocimiento sobre una situación no sea completo (podrán añadirse datos posteriormente), a que la fuente de conocimiento consultada no sea totalmente fiable, a que el conocimiento venga expresado en términos vagos o imprecisos como «grande», «casi», «es posible», etc. Esto hace que las técnicas basadas en la lógica clásica no nos sean útiles para razonar en estas situaciones, sino que tenemos que utilizar técnicas de razonamiento aproximado. En este trabajo pretendemos describir brevemente algunos de los métodos de carácter cualitativo que se utilizan para razonar con incertidumbre, concretamente describiremos el razonamiento plausible y los métodos basados en la no monotonía.

El principio fundamental del razonamiento basado en la lógica clásica consiste en suponer que las creencias del razonador son ciertas, y también en suponer que la verdad nunca cambia. Por tanto, la principal necesidad para razonar será el añadir nuevas creencias al conjunto actual. Sin embargo, al utilizar tal enfoque de carácter monótono nos encontramos con que problemas como los siguientes no pueden ser resueltos:

- Problema del razonamiento basado en el sentido común.
- Problema del marco.

- Problema de control.

El primer problema hace referencia al hecho de que 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 el sentido común es no-monótono.

El problema del marco es uno de los más antiguos en Inteligencia Artificial, y hace referencia al hecho de que en la vida real, en el transcurso de una acción en un sistema dado, la mayoría de la descripción del estado del sistema no varía. Por ejemplo, si el sistema fuera una sala de televisión, tendríamos que el estado del mobiliario siempre sería el mismo, siendo la presencia o ausencia de las personas lo que variaría. El problema está en que describir la parte que no cambia es muy complicado. Sin embargo, esto debe hacerse para un programa de ordenador, ya que él no puede suponer por sí mismo lo que ha sucedido: La situación debe serle explicada.

Por último, el problema de control hace referencia a cómo tomar decisiones teniendo en cuenta las creencias actuales del razonador. Ya que el razonador puede incluso introducir sus deseos e intenciones en el conjunto de creencias. Las *reglas de inferencia* son reglas de modificación de las creencias del razonador. Algunos métodos de no-monotonía tratan este problema.

A continuación se discuten brevemente dos enfoques de carácter cualitativo que permiten realizar razonamiento aproximado, como son el razonamiento plausible y las técnicas basadas en la no monotonía.

1. NO MONOTONÍA

El razonamiento no monótono es muy apropiado cuando se da alguna de las siguientes situaciones:

- el conocimiento que se posee es incompleto
- el universo de discurso es cambiante
- 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 Inteligencia Artificial, 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. A continuación veremos distintos enfoques basados en la no monotonía.

1.1. No monotonía y Lógicas Autoepistémicas

En las lógicas monótonas, si tenemos dos conjuntos de axiomas A y B, tal que $A \subseteq B$, esto implica que $Th\ (A) \subseteq Th\ (B)$, donde $Th\ (A)$ denota el conjunto de los teoremas de A. Esta afirmación no siempre es cierta en las lógicas no monótonas.

La primera contribución importante al desarrollo de las lógicas no monótonas fue hecha por McDermott y Doyle en 1980, quienes modificaron el cálculo de predicados de primer orden introduciendo un nuevo operador proposicional M, cuyo significado era *consistente con la teoría*.

El ejemplo más popular del razonamiento no monótono encontrado en la bibliografía trata sobre la capacidad de volar o no de un pájaro (Tweety en concreto). La capacidad de volar del pájaro se describe en el sistema de McDermott y Doyle de la siguiente forma:

$$(\forall x)\ (pájaro\ (x) \wedge M\ puede\_volar\ (x) \rightarrow puede\_volar\ (x)),$$

o informalmente, «para todo x, si x es un pájaro y es consistente decir que x puede volar, entonces x puede volar» . Este planteamiento de la situación, salva casos como «x es un pingüino».

En 1985 R. C. Moore señaló que el sistema de McDermott y Doyle podía tener una única regla de inferencia no monótona con el siguiente significado intuitivo:

«$M\ P$ es derivable si $\neg P$ es no derivable».

Esto nos lleva a ver que la lógica definida por McDermott y Doyle tiene una noción de consistencia bastante débil, ya que $M\ P$ no es inconsistente con $\neg P$, es decir, P puede ser consistente con la teoría y P puede ser falsa.

También en 1985 Moore observó que en el sistema planteado por McDermott y Doyle se confundían dos tipos distintos de razonamiento no monótono: razonamiento por defecto y razonamiento autoepistémico. El ejemplo visto anteriormente pertenecería al razonamiento por defecto, ya que en ausencia de información que diga lo contrario, la conclusión resultante sería que un pájaro particular (p.e. Tweety) puede volar. Sin embargo, la sentencia «$M\ puede\_volar\ (x)$» sacada también del ejemplo anterior, puede ser interpretada de manera diferente: Los únicos pájaros que no pueden volar son aquellos de los que puede ser inferido que no vuelan. Así, la regla de inferencia para M será, «$M\ P$ es derivable si $\neg P$ no es derivable». Este tipo de razonamiento no es una forma de razonamiento por defecto. En cambio, es un razonamiento basado en nuestras propias creencias o conocimientos, el cual fue llamado *razonamiento autoepistémico* por Moore. El razonamiento autoepistémico difiere del razonamiento por defecto en que en el segundo las conclusiones que se formulan son tentativas, y por tanto pueden

ser retiradas cuando se disponga de mejor información. Esto no puede hacerse en el razonamiento autoepistémico. Por ejemplo, si creemos que todas las clases de pájaros que no pueden volar son conocidas, entonces las conclusiones que obtenemos son finales. El razonamiento autoepistémico no es monótono porque el significado de una sentencia autoepistémica es sensible al contexto, depende de la teoría.

A modo de ejemplo Moore consideró las dos siguientes teorías:

- Teoría 1.
  - Axiomas:

    pájaro (*Tweety*),
    $(\forall x)$ (*pájaro* $(x) \wedge M$ *puede_volar* $(x) \rightarrow$ *puede_volar* (x)).

  - Conclusión: *puede_volar* (*Tweety*) es un teorema de esta teoría.
- Teoría 2.
  - Axiomas:

    $\neg$*puede_volar* (*Tweety*),
    pájaro (*Tweety*),
    $(\forall x)$ (*pájaro* $(x) \wedge M$ *puede_volar* $(x) \rightarrow$ *puede_volar* (x)).

  - Conclusión: *puede_volar* (*Tweety*) no es un teorema de esta teoría.

1.2. Lógica por Defecto

El razonamiento por defecto es una forma bastante económica de rellenar las ranuras en los marcos, es por esto que ha sido muy utilizado en lenguajes y programas de representación del conocimiento basados en marcos. Frecuentemente, el razonamiento por efecto está oculto bajo el nombre de *hipótesis de mundo cerrado*, concepto éste muy utilizado en bases de datos, donde una búsqueda no satisfecha significa la no existencia del item buscado.

Formalmente, una *teoría por defecto* $\Delta = (D, W)$, consiste en un conjunto D de «reglas por defecto» y un conjunto W de fórmulas bien formadas (fbf) del cálculo de predicados de primer orden. Una regla por defecto es una expresión de la siguiente forma:

$$\frac{A : M\,B_1, M\,B_2, ..., M\,B_m}{C},$$

donde A, B_1, B_2, ..., B_m y C son fbf. A se conoce como *precondición*, B_1, ..., B_m como *justificaciones*, y C como la *consecuencia* de la regla. La interpretación intuitiva es la siguiente: «si existe la creencia de A y

cada $B_1, ..., B_m$ puede ser creído consistentemente, entonces podemos creer C. W es un conjunto de axiomas de Δ».

El ejemplo visto anteriormente puede expresarse como sigue mediante una regla por defecto:

$$\frac{pájaro\ (x) : M\ puede\_volar\ (x)}{puede\_volar\ (x)},$$

donde el significado sería: «si x es un pájaro y es consistente asumir que x puede volar, entonces inferiremos que x puede volar».

La regla por defecto más simple que nos podemos encontrar es

$$\frac{: M\ B}{B},$$

es decir, aquella en la que no existe precondición y además la consecuencia es igual a la justificación. Su interpretación es: «si B es consistente con todo lo conocido hasta ahora, entonces podemos creerlo». Los dos tipos de reglas por defecto más importantes son los siguientes:

- regla por defecto *normal*

$$\frac{A : M\ B}{C}$$

- regla por defecto *seminormal*

$$\frac{A : M\ (C \wedge D)}{C}$$

Las reglas por defecto se usan como reglas de inferencia no monótona junto con aquellas que están expresadas en lógica de primer orden. El conjunto de todas las creencias que se pueden observar de una teoría por defecto se conoce como una *extensión* de la teoría por defecto. El concepto de una extensión es muy importante para la lógica por defecto, ya que una extensión representa un conjunto de creencias que se justifican mediante el conocimiento que tenemos sobre un mundo. Las teorías por defecto que únicamente contienen reglas por defecto normales, siempre tienen extensiones. Esto no siempre es cierto cuando las teorías consisten de reglas por defecto seminormales.

1.3. Circumscripción

El motivo para el uso de la circumscripción viene de realizar la siguiente observación en el mundo real: frecuentemente es necesario realizar demasiadas suposiciones para tomar una decisión o para aceptar una creencia (creer algo como cierto). La circumscripción es una regla de inferencia para computar un conjunto de axiomas. En la *cir-*

cumscripción de predicados presentada por McCarthy en 1980, circumscribir una fbf A con respecto a un predicado P contenido en A, es sentenciar que los únicos objetos que satisfacen P son aquellos que se tienen en las bases de A.

Sea P un predicado y sea A (P) una fbf del cálculo de predicados de primer orden, que contiene a P. A (Φ) representa al resultado de sustituir todas las ocurrencias de P en A por una fbf Φ. La circumscripción de predicados de P en A (P) tiene el siguiente esquema:

$$(A\ (\Phi) \wedge (\forall x)\ (\Phi\ (x) \rightarrow P\ (x))) \rightarrow (\forall x)\ (P\ (X) \rightarrow \Phi\ (x)),$$

donde $x = (x_1, x_2, ..., x_n)$, $n \geq 1$. Esto puede interpretarse de la siguiente forma: «los objetos descritos por P y necesarios para satisfacer A (P) son los únicos que son necesarios». La proposición que puede ser obtenida mediante la circumscripción de P en A (P) es un resultado de *inferencia circumscriptiva*. La inferencia circumscriptiva es un ejemplo de razonamiento no monótono.

Por ejemplo, sea P (x) el predicado es_enano (x) y sea A (P) la siguiente fbf

$$es\_enano(Bash\,ful) \wedge es\ enano(Doc) \wedge es\_enano(Dopey) \wedge es\_enano(Grumpy) \wedge es\_enano(Happy) \wedge es\_enano(Sleepy) \wedge es\_enano(Sneezy)$$

Por tanto Bashful, Doc, Dopey, Grumpy, Happy, Sleepy y Sneezy son descritos por P y son necesarios para satisfacer A (P). La circumscripción del predicado es_enano en la fórmula A (P) produce:

$$(\Phi\ (Bashful) \wedge \Phi\ (Doc) \wedge \Phi\ (Dopey) \wedge \Phi\ (Grumpy) \wedge \Phi\ (Happy) \wedge \Phi\ (Sleepy) \wedge \Phi\ (Sneezy) \wedge (\forall x)\ (\Phi\ (x) \rightarrow es\_enano\ (x))) \rightarrow (\forall x)\ (es\_enano\,(x) \rightarrow \Phi\ (x)).$$

Tomemos Φ (x) como la siguiente fórmula:

$$(x = Bashful) \vee (x = Doc) \vee (x = Dopey) \vee (x = Grumpy) \vee (x = Happy) \vee (x = Sneezy) \vee (x = Sleepy).$$

Esto hace que la parte izquierda de la implicación anterior sea cierta, por tanto,

$$(\forall x)\ (es\_enano\,(x) \rightarrow \Phi\ (x)).$$

Así, la conclusión obtenida es que los únicos enanos son Bashful, Doc, Dopey, Grumpy, Happy, Sleepy y Sneezy. La inferencia circumscriptiva del ejemplo es no monótona, ya que si adjuntamos es_enano (Krasnal) a A (P) la conclusión del ejemplo no permanece cierta.

1.4. Sistemas de Mantenimiento de la Verdad (TMS)

Una de las principales contribuciones al desarrollo de los TMS (también llamados «sistemas de revisión de creencias» o «sistemas de mantenimiento de la razón») fue hecha por J. Doyle en 1979. El sistema presentado por Doyle está basado en las justificaciones y tiene dos componentes principales: un resolutor de problemas (el cual realiza inferencias en forma de justificaciones), y un sistema de mantenimiento de la verdad (el cual registra las justificaciones, hace inferencias no monótonas, y chequea en busca de contradicciones). El resolutor de problemas está relacionado con el conocimiento del dominio. El TMS utiliza un procedimiento de satisfacción de restricciones, llamado mantenimiento de la verdad, que a través de inferencias no monótonas determina el conjunto de creencias. Las contradicciones no son toleradas por el TMS, sino que son modificadas mediante la inserción de justificaciones adicionales en un procedimiento especial de *bactraking dirigido por las dependencias*. Por tanto, los datos siempre son consistentes (Fig. 1).

El TMS original fue posteriormente extendido (J. de Kleer, 1986) a un «sistema de mantenimiento de la verdad basado en suposiciones» o ATMS (assumption-based TMS). Este sistema extendido se basa en la manipulación de suposiciones en lugar de centrarse únicamente en las justificaciones. En este sistema se evita la mayoría del backtraking, ya que no se eliminan las inconsistencias. Posteriormente (Reiter y De Kleer, 1987) este sistema fue generalizado a un «sistema de manejo de cláusulas».

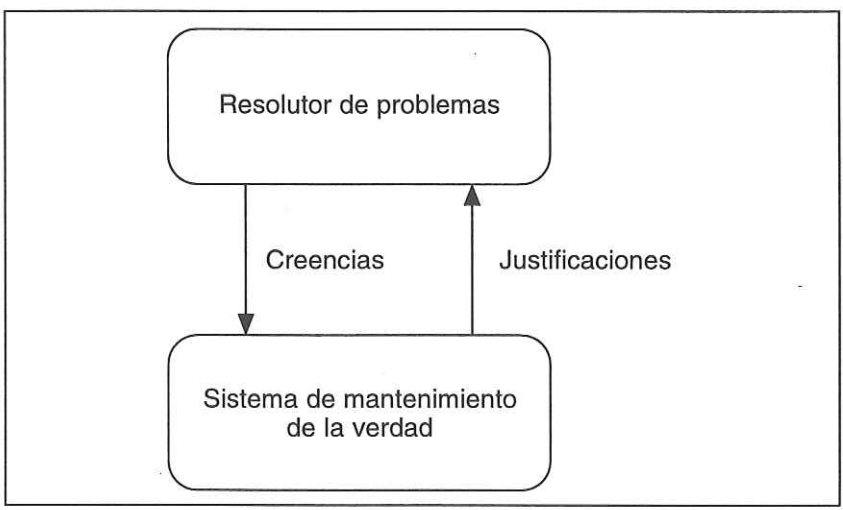

FIGURA 1.
Sistema de Mantenimiento de la Verdad.

En un TMS, existe una estructura de datos especial, llamada **nodo**, la cual representa la creencia que se tiene en una parte del conocimiento del resolutor de problemas, tal como un hecho o una regla de inferencia. Las relaciones en el conjunto de todos los nodos están representados mediante **justificaciones**. Cada nodo tiene asociado un estado soporte «support status». Así como un nodo puede tener varias justificaciones asociadas, sólo unas pocas de ellas serán válidas. Una de ellas es seleccionada como la justificación soporte actual y es llamada *justificación soporte*. Un nodo que no tenga ninguna justificación válida actual, está fuera del conjunto actual de creencias (nodo *out*). El resolutor de problemas puede insertar o borrar nodos y justificaciones, así como marcar ciertos nodos como contradictorios.

Una nueva justificación para un nodo puede provocar que el TMS lo clasifique como nodo *in*, es decir, lo introduzca en el conjunto de creencias. Esto puede provocar a su vez que al invocar el procedimiento de mantenimiento de la verdad, nuevos nodos sean marcados como *in*.

Las contradicciones ocurren cuando un nodo in es marcado como contradictorio, entonces el procedimiento de backtraking dirigido por la dependencia, determina qué suposiciones causan la contradicción. Se añaden justificaciones adicionales al conjunto de «culpables» hasta que la contradicción esté *out* (fuera del conjunto de creencias).

Por tanto, los TMS determinan una asignación de estado de creencia para todos los nodos. Una de las consecuencias de la forma de trabajar de los TMS es que su principal énfasis está en justificar creencias, ignorando la cuestión de verdad.

1.4.1. Justificaciones

Los sistemas de mantenimiento de la verdad utilizan dos tipos distintos de justificaciones para los nodos:

- *«Lista Soporte»*. Tiene el siguiente formato:

  (SL (lista_in) (lista_out)),

  donde lista_in y lista_out son dos listas de nodos. La justificación por lista soporte es *válida* solamente si cada nodo de la lista_in está en IN en el instante actual, y cada nodo de lista_out está en OUT. Por tanto, la creencia de un nodo está basada no sólo en la creencia de otros nodos, si no también en la carencia de creencia en otros nodos, de donde podemos concluir que las justificaciones del TMS son no monótonas. Cuando tanto la lista_in como la lista_out están vacías, el nodo representa una premisa y siempre es válido (nodo siempre a IN). Un nodo cuya lis-

ta_in es no vacía y cuya lista_out es vacía, representa un argumento monótono. Las suposiciones se definen como nodos cuya lista_out no es vacía, representa por tanto una justificación no monótona.

- *«Demostración Condicional»*. Tiene el siguiente formato:

  (CP (consecuente) (hipótesis_in) (hipótesis_out)),

  donde consecuente significa el consecuente de un nodo, y las listas de hipótesis describen la validez de ciertos pros y contras hipotéticos. Una justificación por demostración condicional es *válida* solamente si el nodo consecuente tiene a IN cada nodo de las hipótesis_in y a OUT cada nodo de las hipótesis_out.

El TMS trabaja prácticamente sólo con listas soportes, ya que las justificaciones por demostración condicional son transformadas por el sistema en justificaciones por lista soporte equivalentes.

1.4.2. Tipos de Nodos

Sea el nodo *X* que está a IN, esto es equivalente a decir que el TMS ha seleccionado ya una justificación por lista soporte actual para *X*, llamada *«justificación soporte»*, y que esta justificación es válida. El *«conjunto de nodos soporte»* de *X* es el conjunto de todos los nodos listados en la lista_in y la lista_out de sus justificaciones soporte.

Sea el nodo *X* que está a OUT. En este caso, el *«conjunto de nodos soporte»* de *X* es el conjunto de nodos tal que exactamente un nodo es seleccionado de cada justificación en el conjunto de justificaciones de *X*, o un nodo OUT de la lista_in o un nodo IN de la lista_out. El nodo *X* no puede cambiar su restado a IN, a no ser que al menos uno de sus nodos soporte cambie su estado soporte, o que una nueva justificación válida sea añadida al conjunto de justificaciones de *X*.

El *«conjunto de consecuencias afectadas»* de un nodo *X*, es el conjunto de los nodos tales que contienen a *X* en sus conjuntos de nodos soporte. El *«conjunto repercusión»* de un nodo *X* es la clausura transitiva del conjunto de consecuencias afectadas del nodo *X*.

Veamos un ejemplo que viene dado por la tabla 1 y que representa el sistema de la figura 2.

En la figura, las flechas normales representan listas_in y las flechas grises representan listas_out. Los nodos que tienen asociadas más de una justificación, tienen varias flechas de entrada no agrupadas. El sistema contiene información sobre el estado de una persona. La persona es un estudiante porque la justificación J1 siempre es válida. El conjunto actual de creencias es que la persona es un estudiante graduado en Informática. La información para cada nodo puede verse en la tabla 2.

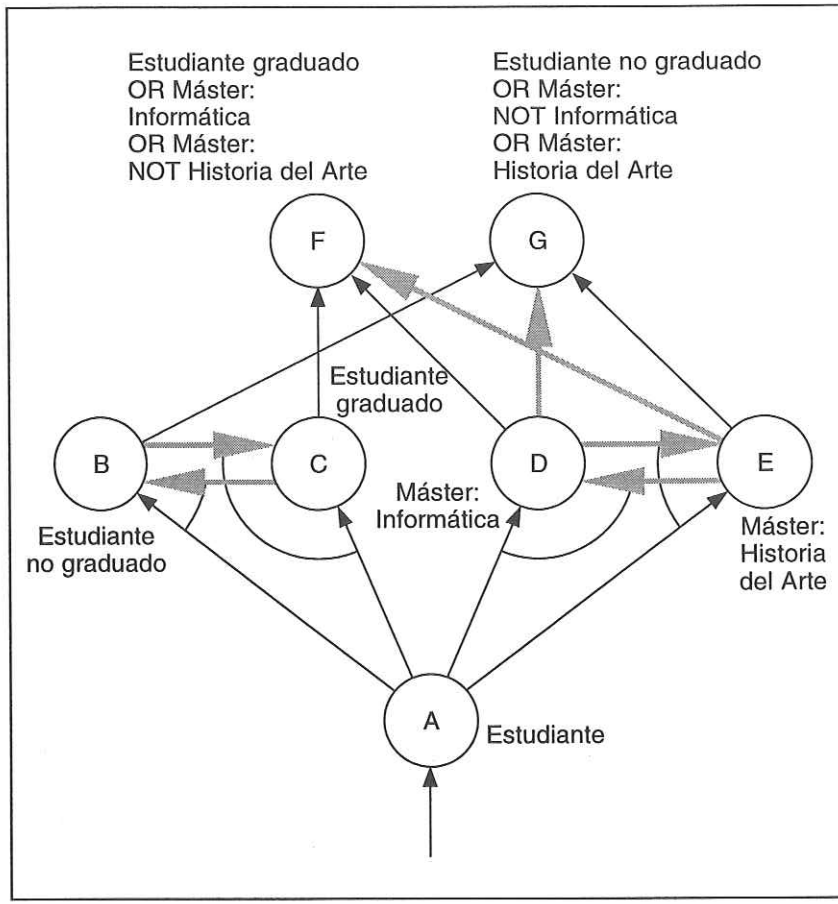

FIGURA 2.
Sistema correspondiente a la tabla 1.

TABLA 1.
Un Sistema de Mantenimiento de la Verdad.

Nodo	Conjunto de justificaciones
A	{ (J1 = (SL ( ) ( )) }
B	{ (J2 = (SL (A) (C)) }
C	{ (J3 = (SL (A) (B)) }
D	{ (J4 = (SL (A) (E)) }
E	{ (J5 = (SL (A) (D)) }
F	{ (J6 = (SL (C) ( )), J7 = (SL (D) ( )), J8 = (SL ( ) (E)) }
G	{ (J9 = (SL (B) ( )), J10 = (SL (E) ( )), J11 = (SL ( ) (D)) }

TABLA 2.
Estado de los conjuntos de nodos.

Nodo	Estado	Justific. S.	Cjto. Nodos S.	Cjto. de Consec.	Cjto. Repercusión
A	in	J1	$\emptyset$	$\{C, D\}$	$\{B, C, D, E, F, G\}$
B	out	ninguna	$\{C\}$	$\{C, G\}$	$\{B, C, F, G\}$
C	in	J3	$\{A, B\}$	$\{B, F\}$	$\{B, C, F, G\}$
D	in	J4	$\{A, E\}$	$\{E, G\}$	$\{D, E, G\}$
E	out	ninguna	$\{D\}$	$\{D, G\}$	$\{D, E, G\}$
F	in	J6	$\{C\}$	$\emptyset$	$\emptyset$
G	out	ninguna	$\{B, D, E\}$	$\emptyset$	$\emptyset$

1.4.3. Mantenimiento de la Verdad

El TMS determina una asignación de estado soporte para todos los nodos. El borrado de una justificación, la inserción de una justificación inválida, o la inserción de una justificación válida a un nodo X cuyo estado soporte es IN no provoca ningún problema. Esencialmente, el problema se produce cuando una nueva justificación válida se añade a un nodo X cuyo estado es OUT. En este caso el nodo y su conjunto repercusión deben ser actualizados. Veamos cómo se hace.

Insertar la nueva justificación en el conjunto de justificaciones del nodo. Añadir el nodo X al conjunto de consecuencias de cada uno de los nodos mencionados en su justificación. Si el nodo X está a in finalizamos, en caso contrario la justificación debe ser chequeada para ver si es válida. si es inválida, añadir un nodo out de su lista_in o un nodo in de su lista_out al conjunto de nodos soporte de X. Si es válida pasar al siguiente paso.

El siguiente paso consiste en chequear el conjunto de consecuencias afectadas por el nodo X. Si es vacío, cambiar el estado del nodo X a in y cambiar el conjunto de nodos soporte de X a la unión de su lista_in y su lista_out. Si el conjunto de consecuencias afectadas no es vacío, crear una lista que contenga al nodo X y a su conjunto repercusión, y marcar su estado como *nil*. Posteriormente se revisan los conjuntos de justificaciones de estos nodos hasta obtener su estado.

Veámoslo con un ejemplo: supongamos que la justificación J12 = (SL ( ) ( )), es añadida al conjunto de justificaciones del nodo B del sistema de la tabla 2. Podemos ver que esta justificación es válida y que el nodo B está a out. El conjunto de consecuencias afectadas del nodo B es distinto del vacío, lo que provoca que se cree la lista $L = \{B, C, F, G\}$. Temporalmente, todos los nodos de L se marcan

TABLA 3.
Nuevo estado de los conjuntos de nodos.

Nodo	Estado	Justific. S.	Cjto. Nodos S.	Cjto. de Consec.	Cjto. Repercusión
A	in	J1	ø	$\{C, D\}$	$\{D, E, F\}$
B	in	J12	ø	$\{C, G\}$	$\{C, G\}$
C	out	ninguna	$\{B\}$	$\{B, F\}$	ø
D	in	J4	$\{A, E\}$	$\{E, G\}$	$\{D, E, F\}$
E	out	ninguna	$\{D\}$	$\{D, G\}$	$\{D, E, F\}$
F	in	J7	$\{D\}$	ø	ø
G	in	J9	$\{B\}$	ø	ø

como *nil*. Comenzamos a mirar las justificaciones y vemos que J12 es válida, lo que hace que B tenga estado in. Pasamos a mirar las justificaciones de C y vemos que J3 es inválida, lo que hace que pase a out. Continuamos así para todos los nodos de L. El estado final puede verse en la tabla 3.

2. RAZONAMIENTO PLAUSIBLE

El razonamiento plausible fue introducido por N. Rescher en 1976. Esta teoría presenta un método para manejar los conflictos entre los datos. Se asume que los datos son mantenidos por fuentes no fiables con diferentes grados de plausibilidad. El razonamiento plausible ofrece unas líneas-guía para razonar en tal situación.

El concepto básico es que de un conjunto no vacío $S = \{P_1, P_2, ..., P_n\}$ de *proposiciones plausibles*, cada proposición P_i está mantenida por una fuente (de información) que tiene un *grado de fiabilidad* positivo. A mayor fiabilidad de la fuente, mayor plausibilidad de la proposición. Como resultado, un número entre 0 y 1 es asignado como *grado de plausibilidad* a cada proposición $P \in S$. El grado de plausibilidad para P se denota como $|P|$. Para una proposición P, $|P| = 1$ si y sólo si la proposición P es cierta, verdadera. Si $|P| = 0$, entonces la proposición P no es nada plausible. Todas las proposiciones de S que sean ciertas deben ser consistentes entre ellas. A continuación veremos a través de ejemplos algunas de las reglas que Rescher propuso para manipular la plausibilidad.

- Si más de una fuente mantiene una misma proposición P, entonces la fuente que tenga mayor grado de fiabilidad determinará el grado de plausibilidad de P. Por ejemplo, como podemos ver en

la tabla 4, la proposición $P \wedge Q$ es mantenida por la fuente B con grado de fiabilidad igual a 0.8, y por la fuente C en grado 0.7. Por tanto, el grado de plausibilidad de $P \wedge Q$ es de 0.8.

TABLA 4.
Grados de Fiabilidad.

Fuente	Grado de Fiabilidad	Proposiciones mantenidas
A	0.9	$P \rightarrow R$
B	0.8	$P \wedge Q$, $P \rightarrow R$
C	0.7	P, $P \wedge Q$

- El conjunto S puede ser aumentado con proposiciones derivadas de él mismo, es decir, con las consecuencias obtenidas del propio S. Por ejemplo, si $S = \{P, P \wedge Q, P \rightarrow R\}$, con los grados de plausibilidad dados en la tabla 5, entonces podemos aumentar S con la proposición R, ya que es consecuencia de P y $P \rightarrow R$.

TABLA 5.
Grados de Plausibilidad.

Proposición	Grado de Plausabilidad
P	0.7
$P \wedge Q$	0.8
$P \rightarrow R$	0.9

- El grado de plausibilidad de una proposición que es derivable desde algún subconjunto consistente T, perteneciente al conjunto de proposiones plausibles S, no es menor que el menor grado de plausibilidad de T. Por ejemplo, en el conjunto aumentado anterior el grado de plausibilidad (a partir de la tabla 2) de la proposición derivada R, es $|R| \geq 0.7$, ya que $|P| = 0.7$ y $|P \rightarrow Q| = 0.9$. Esta regla también es válida cuando los datos son inconsistentes. Por ejemplo, a partir de la tabla 6, podemos ver que $S = \{\neg P \vee Q, \neg Q, P, Q\}$ y que los grados de plausibilidad para las proposiciones de S vienen dadas por la tabla 7.

  Además, a partir de $\neg P \vee Q$ y P podemos aumentar S con la proposición Q, cuyo grado de plausibilidad será $|Q| \geq \min(0.9, 0.6) = 0.6$; similarmente, $\neg P \vee Q$ y $\neg Q$ producen que aumentemos S con la proposición $\neg P$, donde $|\neg P| \geq \min(0.9, 0.9) = 0.9$.

TABLA 6.
Grados de Fiabilidad.

Fuente	Grado de Fiabilidad	Proposiciones mantenidas
A	0.9	$\neg P \vee Q$, $\neg Q$
B	0.6	P, Q

TABLA 7.
Grados de Plausibilidad.

Proposición	Grado de Plausibilidad
$\neg P \vee Q$	0.9
$\neg Q$	0.9
P	0.6
Q	0.6

En el ejemplo podemos ver que $|P| = 0.6$ y $|\neg P| = 0.9$, esto no es nada raro, ya que el grado de plausibilidad no es una probabilidad. La plausibilidad tal y como la entendió Rescher es un concepto clasificatorio. Los números son muy útiles para la representación de algunos grados de plausibilidad pero pueden ser sustituidos por conjuntos de nombres ordenados linealmente, como altamente-plausible, medianamente-plausible, poco-plausible, etc.

- Es posible que una proposición P pueda ser consecuencia de las proposiciones plausibles de S de diferentes formas. Sean

$$P_1^1, P_2^2, \ldots, P_n^1,$$
$$P_1^2, P_2^2, \ldots, P_n^2,$$
$$\ldots$$
$$P_1^i, P_2^i, \ldots, P_n^i,$$

diferentes secuencias de elementos de S, tal que cada fila produce como consecuencia la proposición P. Entonces

$$|P| \geq \max_i \min_j P_j^i.$$

Por ejemplo, sea $S = \{\neg P, P \vee Q, \neg P \rightarrow Q\}$, y $|\neg P| = 0.7$, $|\neg P \rightarrow Q| = 0.5$, y $|P \vee Q| = 0.7$. Existen dos secuencias distintas de los elementos de S que producen la proposición Q:

$$\neg P, \neg P \rightarrow Q$$

y

$$\neg P, P \vee Q.$$

Por tanto,

$$|Q| \geq \max(\min(|\neg P|, |\neg P \rightarrow Q|), \min(|\neg P|, |P \vee Q|)) =$$
$$= \max(\min(0.8, 0.5), \min(0.8, 0.7)) =$$
$$= \max(0.5, 0.7) = 0.7.$$

- El aumentar un conjunto inconsistente S de proposiciones plausibles puede causar que tengamos que revisar los grados originales de pausibilidad de las proposiciones de S, si éstas son añadidas como consecuentes. Por ejemplo, $S = \{P, Q, \neg Q\}$, y $|P| = 0.6$, $|Q| = 0.7$ y $|\neg Q| = 0.7$. Podemos ver que la proposición Q ocasiona $P \vee Q$, y por tanto $|P \vee Q| \geq 0.7$. Ahora, $\neg Q$ y $P \vee Q$ ocasionan la proposición P. Así, por un lado tenemos que $|P| = 0.6$ y por otro que $|P| \geq \min(|\neg Q|, |P \vee Q| = 0.7$, lo que nos lleva a una contradicción. Por lo tanto, el valor original de P debe ser revisado.
- La *regla de neutralización* dice que si el conjunto S de proposiciones plausibles es aumentado a un conjunto S^+ de proposiciones plausibles, y si en S^+ existe un subconjunto consistente T tal que una proposición P «claramente imposible» pudiera ser derivada desde T, entonces S^+ debería contener una proposición P con un grado de plausibilidad tan alto que produjera una ventaja sobre P. La información de que P es «claramente imposible» es dada por un experto. Por ejemplo, sea S el siguiente conjunto de proposiciones plausibles:

  S = {Tweety puede volar, ¬(Tweety es un pingüino), ¬(Tweety es un avestruz), (Tweety es un pingüino) ∨ (Tweety es un avestruz)},

  y una fuente con un grado muy alto de fiabilidad mantiene la proposición (Tweety es un pingüino) ∨ (Tweety es un avestruz). Entonces «Tweety puede volar» es claramente imposible, por tanto la proposición ¬(Tweety puede volar) debería ser añadida a S con un altísimo grado de plausibilidad.
- Una de las tareas más importantes del razonamiento plausible es la conversión de conjuntos inconsistentes de proposiciones plausibles en conjuntos consistentes. Sea S un conjunto de proposiciones plausibles, consistente o no. Un subconjunto no vacío T de S se llama *subconjunto consistente maximal de S* si y sólo si es consistente y además ningún elemento de $S - T$ puede añadirse a T sin producir una inconsistencia.

  Por ejemplo, si $S = \{P, Q, \neg Q, P \vee Q\}$, entonces existen dos subconjuntos consistentes maximales de S. $\{P, Q, P \vee Q\}$ y $\{P, \neg Q, P \vee Q\}$. Entre ambos puede que uno sea más deseable a causa de su plausibilidad. Este procedimiento fue introducido por Rescher para la selección de *consistencias plausibilísticas*.

Si un subconjunto consistente maximal T es obtenido mediante el rechazo de alguna proposición que tuviera un grado de plausibilidad muy alto, entonces T no es elegido como consecuencia plausibilística. Por ejemplo, si en el caso anterior $|P| = 0.8$, $|Q| = 0.2$, $|\neg Q| = 0.9$, y $|P \vee Q| = 0.6$, entonces $\{P, \neg Q, P \vee Q\}$ es la consecuencia plausibilística.

3. CONCLUSIONES

En este trabajo hemos tratado a nivel divulgativo dos modelos para realizar razonamiento cuando los datos son inexactos o nos encontramos en una situación de incertidumbre. Ambos modelos permiten mayor flexibilidad que los basados en la lógica clásica, ya que se pueden acomodar nuevos datos y retraer de la base de conocimiento teoremas que creíamos ciertos, pero que al incorporar nuevo conocimiento han dejado de serlo. Obviamente esta mayor flexibilidad tiene un precio, y éste es el aumento de la complejidad computacional en los sistemas que incorporan no monotonía o razonamiento plausible.

4. BIBLIOGRAFÍA

CASTRO, J. L. y ZURITA, J. M. (1992): «A fuzzy Logic ATMS. Non-Classical Logics and their Applications». *14 Linz Seminar on Fuzzy Set Theory*. Austria.

CLARKE, M., KRUSE, R. y MORAL, S. (1993): «Symbolic and Quantitative Approaches to Reasoning and Uncertainty». *Lecture Notes in Computer Science*. Springer-Verlag.

DOYLE, J. (1979): «A truth maintenance system». *Artificial Intelligence*. 12 (3).

FROST, R. (1989): *Bases de Datos y Sistemas Expertos*. Díaz de Santos.

GALE, W. A. (1986): *Artificial Intelligence & Statistics*. Addison Wesley.

MCCARTY, J. (1977): «Epistemological problems of Artificial Intelligence». *IJCAI-5*, 1038-1044.

MCCARTHY, J. (1980): «Circumscription - a form of non-monotonic reasoning». *Artificial Intelligence*, 13 (1), 27-40.

MCDERMOTT, D. V. y DOYLE, J. (1980): «Non-monotonic logic I». *Artificial Intelligence*. 13 (12), 41-72.

NILSSON, N. J. (1987): *Principios de Inteligencia Artificial*. Díaz de Santos.

REITER, R. (1978): «On reasoning by default». *TINLAP-2*, 210-218.

RESCHER, N. (1976): *Plausible reasoning*. Amsterdam: Van Garcum.

RICH, E. y KNIGHT, K. (1995): *Inteligencia Artificial*. McGraw-Hill.