ARTÍCULO

Una forma normal temporal independiente del método de deducción

Manuel Enciso, Inmaculada P. de Guzmán, Carlos Rossi *

E.T.S.I. Informática.
Universidad de Málaga
Campus de Teatinos. 29071 Málaga, Spain
enciso@lcc.uma.es, {guzman, rossi}@ctima.uma.es

Resumen

En este trabajo se presenta la definición de una forma normal para una lógica modal temporal. Como lógica objetivo hemos elegido la lógica LNint-e, que destaca por tratarse de una lógica que combina puntos e intervalos y disponer simultáneamente de un tratamiento del tiempo absoluto y relativo.
Nuestro principal objetivo al definir la forma normal ha sido dotarla de una absoluta independencia respecto al método de deducción. De este modo, dicha forma normal puede ser usada por cualquier demostrador automático de teoremas, con diferentes paradigmas (resolución, tablas semánticas, etc.) e incluso ser incorporada a métodos de deducción como el chequeo de modelos para aumentar el rendimiento de éstos. Así, en el proceso de normalización se busca esencialmente reducir la complejidad de las expresiones lógicas en lugar de preparar ésta para un determinado tratamiento automático.
La definición introducida se basa principalmente en el concepto de *literal temporal*. Este concepto permite la construcción de transformaciones de equivalencia que reducen la complejidad de la fórmula de entrada. Entre estas transformaciones conviene destacar por su novedad aquellas que permiten la eliminación de conectivas temporales binarias, que suponen un mayor coste de procesamiento en los métodos de pruebas. Los beneficios de este trabajo han sido comprobados empíricamente mediante una implementación del método, cuyos resultados se comentan también en este trabajo.

Palabras clave: Razonamiento temporal, lógica, forma normal.

1. Introducción

El uso de las técnicas de normalización como base para el tratamiento automático de expresiones ha sido una constante en el desarrollo previo de las técnicas de deducción y tratamiento automático. En ciertas áreas la normalización supone un punto y aparte en su desarrollo, como en el caso del diseño de bases de datos relacionales o la programación lógica. Sin embargo el número de trabajos que se ha dedicado a la normalización de expresiones de lógicas temporales es muy bajo y, si nos centramos en lógicas modales temporales con cierto interés computacional, podemos decir sin temor a equivocarnos[1] que ha sido abordado

*Este trabajo ha sido parcialmente subvencionado por el proyecto CICYT TIC-2000-1109.

[1]En los datos obtenidos de la bibliografía de la Universidad de Trier (`http://dblp.uni-trier.de`) aparecen 202 referencias en relación con la normalización de expresiones (formas normales en lógica clásica, multivaluada, lógica de reescritura, etc), formas normales en entorno de tratamiento de matrices, formas normales en bases de datos, etc. Sin embargo, las referencias

prácticamente en exclusiva por el grupo liderado por el profesor M. Fisher de la Universidad de Liverpool.

El interés de este activo y prestigioso grupo de investigación por este tema lo podemos encontrar en una cita de uno de sus trabajos: *una de las barreras para el desarrollo de métodos de resolución clausal para lógicas modales y temporales ha sido la dificultad encontrada en la producción de una forma normal para dichas fórmulas.* Desde este prisma, una vez encontrada la forma normal, el desarrollo de técnicas de deducción automática para las lógicas modales y temporales sería una tarea más sencilla y directa. Sin embargo, esta conclusión no cuadra con el escaso interés que ha mostrado la comunidad científica por el desarrollo de formas normales temporales en contraposición con el interés mostrado en la búsqueda de métodos de prueba automatizables.

En nuestra opinión este aparente divorcio entre forma normal y métodos de deducción se debe a varias razones: además de la diversidad de lógicas modales y temporales, debemos apuntar la no existencia de un método de prueba universalmente aceptado para estas lógicas[2].

Si nos centramos en lógicas temporales (o combinaciones con ellas) que incluyan conectivas binarias (*Until* o *Since*), podemos citar como primer trabajo relevante a [13], donde el autor presenta una *forma normal separada* para la lógica temporal totalmente expresiva sobre tiempo lineal discreto.

Su principal objetivo es preparar a la fórmula para ser tratada usando el método de resolución temporal desarrollado en [12]. Se obtiene una fórmula que separa las expresiones de pasado, futuro y presente. Dichas expresiones están basadas en tres componentes básicas descritas en forma de reglas: una regla *inicial*, una $\Box$-regla y una $\Diamond$-regla que se unen de modo que la forma normal tiene la forma $\Box\wedge^{n}_{i=1}(P_i \rightarrow F_i)$ donde P_i es una expresión de presente-pasado y F_i es una expresión de presente-futuro.

La forma normal así definida tiene efectos muy beneficiosos para la prueba final, pero en el proceso de transformación, además de oscurecer en exceso la lectura de la especificación inicial, se produce un aumento del tamaño de la fórmula y, en algunos casos, aparecen nuevos símbolos de variable por renombramiento de subexpresiones (esta técnica fue introducida en [26] para la normalización en lógicas temporales).

Dicho trabajo ha sido extendido notablemente en [9] presentando un método de prueba para una combinación de una lógica modal (KD45) y la lógica temporal ramificada CTL. El método de prueba requiere la conversión a una forma normal que separa las componentes dinámicas de las de creencia.

Esta forma normal está guiada por el propio método de pruebas y se aprecia en la caracterización de las componentes sobre las que está basada. Estas componentes establecen una relación, mediante la conectiva $\rightarrow$, entre claúsulas (de diversos tipos) y constantes (*start* o *true*) o fórmulas monarias (usando la conectiva temporal $\Diamond$ o las de creencia E o A).

Se puede constatar la fuerte relación entre los dos trabajos citados al observar cómo las formas normales conservan la misma orientación en lo que se refiere al tratamiento temporal: forma normal basada en componentes de tipo regla, aumento de la longitud de las expresiones en el proceso de normalización e inclusión de nuevos símbolos de variable para el renombramiento de subexpresiones.

En este trabajo presentamos una orientación diferente para el problema de la normalización de lógicas temporales totalmente expresivas. Creemos que se hace necesario la introducción de una forma normal que no busque la adaptación directa a un cierto método de deducción (como las introducidas hasta ahora), sino más bien la existencia de una forma normal *independiente* cuya principal motivación sea la simplificación de las expresiones de la lógica para disponer de una expresión *canónica*. Así, en este trabajo presentamos una forma normal cuyas principales características son:

- No está orientada a un tratamiento posterior por ninguno de los métodos de deducción automáticos conocidos, sino que está guiada por la propia semántica de las conectivas temporales y sus interrelaciones.
- Propone el uso de *literales temporales* en contraposición al uso de componentes básicos tipo reglas. Este concepto permite el di-

sobre formas normales en lógica temporal sólo suponen el 1,48 %.

[2] La fuerte relación existente entre la forma normal y el método de prueba hace que los resultados obtenidos para la primera no sean aprovechables cuando cambiamos de método de deducción.

seño de transformaciones de simplificación muy eficientes y con coste lineal.

- A partir de un detallado estudio del coste de las subexpresiones que componen una fórmula temporal, se ha realizado un diseño general de reducción de las expresiones para permitir un tratamiento automático más eficiente (independientemente del método elegido). En particular se han incluido transformaciones que reducen expresiones con conectivas temporales binarias a otras equivalentes con conectivas monarias, que en todos los casos supone una ganancia en el coste final de la demostración.

Para mostrar además de la independencia con respecto al método de deducción, la flexibilidad con respecto a las especificaciones iniciales, el trabajo no toma como punto de partida una lógica de puntos al uso. Se ha elegido como lógica original LNint-e, una lógica de intervalos basada en la lógica de puntos LN [3]. LNint-e (que extiende el trabajo presentado en [7]) es una lógica sobre tiempo discreto con total potencia expresiva, que combina las aproximaciones absoluta y relativa. Por otra parte, LNint-e tiene una semántica topológica que como se mostrará más adelante, es muy adecuada para Computación. Este hecho queda confirmado por la utilidad para aplicaciones de LNint-e [10, 11].

Los resultados aportados en este trabajo ilustran cómo allanar el camino para el tratamiento de las expresiones de intervalos desde los métodos de deducción basados en lógicas temporales de puntos. Así, podemos hablar de un uso futuro de métodos *semi-directos* y no indirectos, como se acostumbra a usar para la lógica de intervalos [16].

En las dos siguientes secciones se describen formalmente la sintaxis y la semántica de LNint-e, mientras que en la sección 4 se muestran algunos resultados sobre su potencia expresiva, tanto para expresiones de puntos como de intervalos. En las secciones 5 y 6 se presentan, respectivamente, ideas introductorias sobre la definición de formal normal buscada, así como algunos resultados previos necesarios para afrontar en las secciones 7 y 8 las dos tareas esenciales en dicha definición, como son la elección de literales y las transformaciones de reducción. Posteriormente, en la sección 9 se establece la definición formal de forma normal negativa para LNint-e. Finalmente, se presentan las principales conclusiones de este trabajo y los tareas cuyo desarrollo nos planteamos en el futuro. Entre las primeras hemos de destacar la inclusión de los resultados obtenidos tras la implementación del algoritmo de normalización y su aplicación a una batería de fórmulas de prueba.

2. La lógica LNint-e

Como hemos indicado, LNint-e es una lógica de puntos extendida para hablar de intervalos, directamente basada en la lógica LN (introducida en [3]). LN es una lógica modal temporal de puntos sobre tiempo lineal discreto e infinito, equivalente a la lógica US [21] (es decir, es totalmente expresiva). Nuestra elección de la lógica LN es debida a varias razones, entre las que destacamos su semántica topológica, fuertemente basada en el flujo del tiempo, que evita el uso de la lógica de primer orden como metateoría. De este modo, las demostraciones se simplifican considerablemente y, en consecuencia, se facilita nuestro trabajo. Otra razón importante es la naturalidad de sus conectivas, que recogen los conceptos de precedencia y simultaneidad. Las conectivas de LN se refieren a la ubicación temporal de las próximas (o últimas) apariciones de sus argumentos. Las conectivas temporales de LN, denotadas $\preccurlyeq$ y $\succcurlyeq$, tienen el siguiente significado intuitivo:

$A \preccurlyeq B$ se lee *alguna vez en el futuro A ocurrirá, y la próxima aparición de A será anterior o simultánea a la próxima aparición de B.*

$A \succcurlyeq B$ se lee *alguna vez en el pasado ocurrió A, y la última aparición de A fue posterior o simultánea a última aparición de B.*

Al estar basada en LN, la lógica LNint-e hereda algunas de sus características, como el flujo del tiempo, la semántica topológica y las conectivas básicas. Además, LNint-e tiene características particulares, entre las que destacamos las siguientes:

- LNint-e tiene total potencia expresiva en cuanto a expresiones de puntos dado que incluye a la lógica LN; con respecto a expresiones de intervalos tiene al menos la misma potencia expresiva que las lógicas de Allen [1] y Halpern y Shoham [17]. Además, LNint-e permite la especificación de relaciones entre cualquier número de intervalos.
- Permite la posibilidad de tratar el tiempo tanto de forma absoluta (vinculando aser-

ciones con fechas o instantes conocidos) como relativa (expresando la relación temporal entre varias aserciones). Esta posibilidad no es contemplada tradicionalmente en las lógicas modales temporales.

- Se diferencia entre *clases* de eventos y las diferentes *ejecuciones* (o *tokens*) de cada una de ellas.

2.1. Sintaxis de LNint-e

A nivel sintáctico, comenzamos considerando una componente para recoger aserciones acerca de puntos y fechas. Hemos de destacar que las expresiones de puntos pueden ser afirmadas sobre intervalos, en cuyo caso hablaremos de expresiones *hereditarias* de intervalos.

Para completar el tratamiento de intervalos, necesitamos otra componente para recoger eventos. Nos referiremos a los eventos, en el sentido de Allen [1], es decir, expresiones sobre intervalos que no son ciertas en los subintervalos ni, más específicamente, en los puntos del intervalo sobre el que se afirma. Las expresiones de eventos serán también llamadas expresiones *no hereditarias* de intervalos. Como se indicó anteriormente, en LNint-e distinguiremos entre *clases* de eventos y *ejecuciones* de eventos, que representarán cada una de las apariciones concretas de una clase de eventos [3]. Esta técnica ha sido empleada anteriormente por varios autores (véase, por ejemplo, [1], [25] y [18]) y nos permite expresar situaciones temporales de contigüidad y solapamiento de ejecuciones de un mismo evento.

En este trabajo trataremos exclusivamente el fragmento de futuro de LNint-e, que denominamos LNint-e$^+$. Las definiciones y resultados para el fragmento de pasado se obtienen por la imagen del espejo[4]. A continuación presentamos formalmente la sintaxis y semántica de LNint-e$^+$.

El lenguaje de LNint-e$^+$, denotado $\mathcal{LN}^+_{int-e}$, tiene como alfabeto el conjunto de símbolos

$$\begin{aligned}\Omega^{ev\text{-}e}_{ins} \;=\; & \Omega_{ins} \cup \underline{\mathbb{Z}} \cup \{\top, \bot\} \cup \\ & \{\uparrow\alpha_e, \downarrow\alpha_e, \overrightarrow{\alpha_e} \mid \alpha_e \in EjecEventos\} \\ & \cup \{\uparrow\alpha_*, \downarrow\alpha_*, \overrightarrow{\alpha_*} \mid \alpha \in \Omega_{ev}\}\end{aligned}$$

donde

- $\Omega_{ins} = \{p, q, \ldots, p_1, q_1, \ldots, p_n, q_n, \ldots\}$ es el conjunto de atómos de puntos. Como hemos indicado anteriormente, estos átomos de puntos también pueden afirmarse sobre intervalos para especificar expresiones hereditarias.
- $\underline{\mathbb{Z}} = \{\underline{m} \mid m \in \mathbb{Z}\}$ es el conjunto de átomos de *fechas*, para nombrar instantes.
- $\Omega_{ev} = \{\alpha, \beta, \ldots, \alpha_1, \beta_1, \ldots, \alpha_n, \beta_n, \ldots\}$ es el conjunto de atómos de clases de eventos.

  Para distinguir las ejecuciones de cada una de estas clases, el alfabeto incluye:

  - $Etiq = \{e, e_1, e_2, \ldots, e_n, \ldots\}$, un conjunto a cuyos elementos denominamos etiquetas. Con esto se define $EjecEventos = \{\alpha_e \mid \alpha \in \Omega_{ev}, e \in Etiq\}$ el conjunto de todas las ejecuciones de eventos.
  - El símbolo $*$, que representa un comodín para etiquetas de ejecución de eventos. De esta manera, en el lenguaje de LNint-e se incluye la posibilidad de construir expresiones en las que se haga referencia a una *ejecución anónima* de un evento dado, de la que, o bien desconocemos la etiqueta concreta, o bien no nos interesa precisarla. Dada una clase de eventos, $\alpha \in \Omega_{ev}$, la notación para una ejecución anónima de α es α_*.
- Los símbolos $\uparrow$, $\downarrow$, $\longrightarrow$, [,], (,) y "," .
- Las conectivas y constantes booleanas $\neg$, $\wedge$, $\vee$, $\top$ y $\bot$.
- La conectiva temporal binaria de puntos $\preccurlyeq$.

En la construcción del lenguaje de LNint-e$^+$ usaremos la técnica de caracterizar los eventos mediante sus instantes de inicio, finalización, y transcurso. Así, para cada ejecución de evento atómico $\alpha_e \in EjecEventos$ usaremos la siguiente notación:

- $\uparrow\alpha_e$ se lee *en este instante comienza la ejecución e del evento* α.
- $\downarrow\alpha_e$ se lee *en este instante finaliza la ejecución e del evento* α.

[3] Hemos de puntualizar que una ejecución dada de una clase de eventos será cierta (a lo sumo) en un único intervalo del flujo del tiempo.

[4] Aunque el resto del trabajo se centre en LNint-e$^+$, gran parte de los resultados que se presentan en este trabajo son también válidos para el fragmento de pasado de LNint-e.

- $\overrightarrow{\alpha_e}$ se lee *en este instante la ejecución e del evento α está transcurriendo.*

Definición 2.1 *El lenguaje $\mathcal{LN}^{+}_{int-e}$ es el cierre inductivo libremente generado sobre el conjunto base $\Omega^{ev\text{-}e}_{ins}$ por la conectiva monaria $\neg$ y las conectivas binarias $\wedge, \vee$ y $\preccurlyeq$* [5].

Una vez que hemos presentado el lenguaje de LNint-e definimos la semántica, que confirmará las lecturas de las conectivas.

3. Semántica de LNint-e$^+$

Emplearemos $(\mathbb{Z}, \leq)$ como flujo del tiempo y para definir la semántica usaremos un concepto clave notado m^{+}_{tA} y definido de la siguiente manera:

Definición 3.1 *Si A es una fórmula bien formada (en adelante fbf) de LNint-e$^+$ y $t \in \mathbb{Z}$, definimos:*

$$m^{+}_{tA} = min\{t' \in \mathbb{Z} \mid t' > t \ \text{ y } A \text{ es cierto en } t'\}$$

Convenimos que $min\, \varnothing = +\infty$. De la misma manera, extendemos el orden de $\mathbb{Z}$ sobre el conjunto $\{+\infty\} \cup \mathbb{Z}$ de la forma habitual.

m^{+}_{tA} se introduce para capturar el hecho de que toda aplicación en Computación está interesada en la posibilidad de decidir cuál es la próxima aparición de un suceso A posterior al instante actual t (desde el que hablamos o ejecutamos).

Denotamos por $Int(\mathbb{Z}) = \{[t_1, t_2] \mid t_1, t_2 \in \mathbb{Z}, t_1 < t_2\}$, el conjunto de intervalos cerrados y finitos de $\mathbb{Z}$ cuyos extremos sean distintos, es decir, intervalos que no sean un punto.

Definición 3.2 *Una* **interpretación temporal** *para LNint-e es una terna*

$$\mathcal{I} = \langle H_{ev}, H_{exec}, h\rangle$$

donde:

- $H_{ev}\colon \Omega_{ev} \longrightarrow 2^{Int(\mathbb{Z})}$ *asocia a cada átomo de clase de evento $\alpha \in \Omega_{ev}$ el conjunto de todos los intervalos en los que se producen las ejecuciones de α.*
- $H_{exec}\colon EjecEventos \longrightarrow Int(\mathbb{Z})$ *asocia a cada ejecución de evento el intervalo de $Int(\mathbb{Z})$ en el que se verifica y cumple las condiciones siguientes:*
  1. $H_{exec}(\alpha_e) \in H_{ev}(\alpha)$ *para todo* $\alpha_e \in EjecEventos$
  2. *Para todo* $\alpha \in \Omega_{ev}$ *y* $e, e' \in Etiq$, *si* $H_{exec}(\alpha_e) = [t_1, t_2]$ *y* $H_{exec}(\alpha_{e'}) = [t'_1, t'_2]$ *se tiene que* $t_2 - t_1 = t'_2 - t'_1$.
- $h\colon Atom^{ev}_{ins} \longrightarrow 2^{\mathbb{Z}}$ *asocia a cada átomo de $\Omega^{ev\text{-}e}_{ins}$ un subconjunto de $\mathbb{Z}$ y cumple las siguientes condiciones:*
  1. $h(\underline{t}) = \{t\}$, *para todo* $\underline{t} \in \underline{\mathbb{Z}}$
  2. $h(\bot) = \varnothing; \quad h(\top) = T$
  3. *Para todo* $\alpha_e \in EjecEventos$, *si* $H_{exec}(\alpha_e) = [t_1, t_2]$, *entonces*
     - *3.1* $h(\uparrow \alpha_e) = \{t_1\}$
     - *3.2* $h(\downarrow \alpha_e) = \{t_2\}$
     - *3.3* $h(\overrightarrow{\alpha_e}) = (t_1, t_2)$
  4. *Para todo* $\alpha \in \Omega_{ev}$, *se tiene que*
     - *4.1* $h(\uparrow \alpha_*) = \{t \mid t \in h(\uparrow \alpha_e), e \in Etiq\}$
     - *4.2* $h(\downarrow \alpha_*) = \{t \mid t \in h(\downarrow \alpha_e), e \in Etiq\}$
     - *4.3* $h(\overrightarrow{\alpha_*}) = \{t \mid t \in h(\overrightarrow{\alpha_e}), e \in Etiq\}$

Esta definición asegura que, como deseábamos, la semántica de las ejecuciones anónimas de eventos recoge cualquier ejecución de la clase de eventos en cuestión. Asimismo, aseguramos la unicidad de las ejecuciones de eventos, es decir, cada ejecución de una clase de eventos, α_e, será cierta en a lo sumo un intervalo,[6] $H_{exec}(\alpha_e)$.

La extensión de una interpretación $\mathcal{I} = \langle H_{ev}, H_{exec}, h\rangle$ a toda fbf de $\mathcal{LN}^{+}_{int-e}$, denotada también $\mathcal{I}$, únicamente requiere la extensión de la función h a $\mathcal{LN}^{+}_{int-e}$, denotada también h, como sigue:

Si $A, B \in \mathcal{LN}^{+}_{int-e}$:

$$\begin{aligned} h(\neg A) &= \mathbb{Z} \setminus h(A) \\ h(A \vee B) &= h(A) \cup h(B) \\ h(A \wedge B) &= h(A) \cap h(B) \\ h(A \preccurlyeq B) &= \{t \in \mathbb{Z} \mid m^{+}_{tA} < +\infty, m^{+}_{tA} \leq m^{+}_{tB}\} \end{aligned}$$

[5] Por consiguiente, $\Omega^{ev\text{-}e}_{ins}$ es el conjunto de todas las expresiones atómicas de nuestro lenguaje que tienen sentido afirmadas sobre puntos.

[6] Obsérvese que, en principio, H_{exec} es una función parcial, ya que pueden existir ejecuciones de eventos que nunca se lleven a cabo.

donde $m^+_{tA} = \text{mín}\left((t, +\infty) \cap h(A)\right)$

Las definiciones de satisfactibilidad, validez, equivalencia y deducción semántica se establecen de la forma habitual.

4. Expresividad de LNint-e

El interés de toda nueva lógica es el equilibrio de su expresividad y la eficiencia computacional de su proceso, tanto en tareas de especificación como en demostradores automáticos.

En esta sección detallamos algunos aspectos sobre la expresividad de LNint-e$^+$, con un objetivo doble: mostrar su potencia expresiva y familiarizar al lector con la construcción de expresiones en nuestra lógica. Diferenciaremos los aspectos relativos a puntos o intervalos y, para cada uno de ellos, los tratamientos absoluto y relativo.

4.1. Expresiones de puntos

La conectiva temporal $\preccurlyeq$ de LNint-e$^+$ expresa una relación de *precedencia no estricta* entre las próximas apariciones de sus dos argumentos. Además, tiene una naturaleza *fuerte* en la terminología de Kröger [23], esto es, para que $A \preccurlyeq B$ sea satisfecha en un instante t, es necesario que A sea cierta en un instante t' posterior a t. Además de $\preccurlyeq$, en LNint-e$^+$ podemos definir de una manera natural (en términos relativos a Computación) conectivas de precedencia estricta y/o débiles. Presentamos a continuación estas conectivas, mostrando su nombre, notación y semántica:

I) *Conectiva fuerte de precedencia estricta (denotada $\prec$)*:

$h(A \prec B) = \{t \in \mathbb{Z} \mid m^+_{tA} < m^+_{tB}\}$

II) *Conectiva débil de precedencia estricta (denotada $\sqsubset$)*:

$$h(A \sqsubset B) = \{t \in \mathbb{Z} \mid m^+_{tB} = \infty\} \cup \{t \in \mathbb{Z} \mid m^+_{tA} < m^+_{tB}\}$$

III) *Conectiva débil de precedencia no estricta (denotada $\sqsubseteq$)*:

$h(A \sqsubseteq B) = \{t \in \mathbb{Z} \mid m^+_{tA} \leq m^+_{tB}\}$

IV) *Conectiva fuerte de simultaneidad (denotada $\approx^+$)*:

$h(A \approx^+ B) = \{t \in \mathbb{Z} \mid m^+_{tA} = m^+_{tB} < +\infty\}$

Es inmediato demostrar que las definiciones en LNint-e$^+$ de las conectivas anteriores son las siguientes:

$$\begin{array}{lcl} A \sqsubset B & \stackrel{def}{=} & \neg(B \preccurlyeq A) \\ A \prec B & \stackrel{def}{=} & A \preccurlyeq B \wedge \neg(B \preccurlyeq A) \\ A \sqsubseteq B & \stackrel{def}{=} & \neg(B \prec A) \\ A \approx^+ B & \stackrel{def}{=} & A \preccurlyeq B \wedge B \preccurlyeq A \end{array}$$

Puesto que LNint-e es totalmente expresiva [27], podemos definir en LNint-e$^+$ conectivas temporales estándar como siguientes:

- FA (leida A será cierto alguna vez en el futuro) se define en LNint-e$^+$ como $FA \stackrel{def}{=} A \preccurlyeq \bot$
- GA (leida A será siempre cierto en el futuro), cuya definición en LNint-e$^+$ es $GA \stackrel{def}{=} \bot \sqsubseteq \neg A$
- $\oplus A$ (leida A será cierto *mañana*), cuya definición es $\oplus A \stackrel{def}{=} A \preccurlyeq \top$
- $U(A, B)$ (leida A será cierto en el futuro y B será cierto desde ahora hasta A), cuya definición es $U(A, B) \stackrel{def}{=} A \preccurlyeq \neg B$
- $A\, atnext\, B$ (leida A será verdadero en la próxima aparición de B), cuya definición es $A\, atnext\, B \stackrel{def}{=} (A \wedge B) \preccurlyeq B$
- $A\, atlast\, B$ (leida A fue verdadero en la última aparición de B), cuya definición es $A\, atlast\, B \stackrel{def}{=} (A \wedge B) \succcurlyeq B$

Además de estas conectivas, y para disponer de la aproximación absoluta para el tratamiento del tiempo definimos:

$$A\, at\, \underline{m} \stackrel{def}{=} (A \wedge \underline{m}) \vee A\, atnext\, \underline{m} \vee A\, atlast\, \underline{m}$$

Con esta conectiva conseguimos saber si una fórmula es verdadera en un instante determinado[7], independientemente del instante desde el que hablemos, es decir, logramos relativizar el instante actual.

[7] Hemos incluido la definición de esta conectiva *absoluta* para ilustrar la potencia expresiva de LNint-e, a pesar de que en ella se emplean conectivas de pasado no incluidas en LNint-e$^+$. Hemos de indicar que en la semántica de esta expresión es necesario el concepto de m^-, definido de forma simétrica a m^+.

4.2. Expresiones de intervalos

Uno de nuestros objetivos es disponer de conectivas explícitas de intervalos. Previamente, es preciso hacer algunas observaciones:

- Al igual que se ha hecho en las expresiones de puntos, distinguiremos, tanto para eventos como para expresiones hereditarias, dos tipos de conectivas: las que dan un tratamiento absoluto y las que dan un tratamiento relativo.
- En LNint-e$^+$, tanto para relaciones entre eventos como para relaciones entre expresiones hereditarias, se sigue el siguiente esquema: las relaciones se construyen en base a un cambio de referencia (con respecto a las lógicas modales de intervalos) en el que se abandona el intervalo actual, para pasar a hablar desde el instante actual y describir desde éste lo que ocurre acerca de relaciones entre intervalos, diferenciando si éstos se dan en el futuro o en el pasado con respecto al instante actual, o en un intervalo que contenga a éste. Esta construcción tiene una consecuencia importante: las conectivas de intervalos se refieren a relaciones temporales entre las próximas (o últimas) apariciones de los intervalos de los que se habla, heredando la filosofía de las relaciones temporales para puntos de LN.

En los apartados siguientes presentamos algunas de estas conectivas.

4.3. Expresiones de eventos

Al construir esta colección de conectivas definidas consideraremos que, como se indicó anteriormente, los eventos pueden ser tratados de dos formas diferentes:

I) Tratamiento de eventos de manera absoluta, es decir, ligando los eventos con intervalos de extremos (fechas) conocidos.

Para ello definimos en $\mathcal{LN}_{int-e}$ una conectiva análoga a la conectiva *at* para el tratamiento absoluto de expresiones de puntos.

Definición 4.1 *Dadas* $\alpha_e \in EjecEventos$ *y* $\underline{m}, \underline{n} \in \underline{\mathbb{Z}}$ *tales que* $m < n$*, definimos*

$$\alpha_e \; at_{ev} \; [\underline{m}, \underline{n}] \stackrel{def}{=} (\uparrow\alpha_e \wedge \downarrow\alpha_e \approx^+ \underline{n}) \; at \; \underline{m}$$

$\alpha_e \; at_{ev} \; [\underline{m}, \underline{n}]$ *se lee* la ejecución α_e sucede exactamente en el intervalo $[m, n]$.

La definición de esta conectiva para ejecuciones anónimas requiere un ligera modificación, completamente documentada en [27].

II) Tratamiento de eventos de manera relativa, expresando la relación entre dos eventos, independientemente de la posición concreta de ambos en la línea del tiempo.

Definimos ahora sólo algunas (por razones de espacio) de las trece posibles relaciones entre eventos [1] en el futuro[8]. Las definiciones referentes a eventos en el pasado se hacen siguiendo la imagen del espejo. Las relaciones que atañen a eventos "actuales" (es decir, que contienen al instante actual) están completamente detalladas en [27].

Dentro de este conjunto de conectivas temporales relativas vuelve a ser necesario considerar dos colecciones de conectivas: una, con definiciones más simples, en la que no se permiten ejecuciones anónimas, y otra, en cierta medida más compleja, en la que si se pueden emplear (véase [27]). Naturalmente, la aplicación en que se usen las conectivas determinará cuál de las dos colecciones ha de utilizarse. A continuación se presentan algunas de las definiciones de la primera colección de conectivas.

4.3.1. Conectivas para eventos en el futuro

A continuación, como ejemplo, se muestran las definiciones en LNint-e$^+$ de las conectivas básicas de simultaneidad de eventos (denotada eq^+) y de vecindad (denotada ab^+). En todas ellas consideraremos que $\alpha_e, \beta_{e'} \in EjecEventos$.

- $eq^+(\alpha_e, \beta_{e'})$ es cierto en un instante t si α_e y $\beta_{e'}$ son ciertos[9] en un intervalo $[t_1, t_2]$, donde $t < t_1 < t_2$.

[8] En esta colección de conectivas exigiremos que los dos eventos relacionados se verifiquen completamente en el futuro, sin contener en ningún caso al instante actual. Relajar esta condición no supondría ninguna dificultad, como se puede observar en [27].

[9] Obsérvese que no es necesario referirse a la *próxima* vez que se dé el evento α_e, ya que las ejecuciones de eventos son únicas en el tiempo.

La definición en LNint-e es: $eq^+(\alpha_e, \beta_{e'}) \stackrel{def}{=} \uparrow \alpha_e \approx^+ \uparrow\beta_{e'} \wedge \downarrow\alpha_e \approx^+ \downarrow\beta_{e'}$

- $ab^+(\alpha_e, \beta_{e'})$ es cierto en un instante t si α_e es cierto en un intervalo $[t_1, t_2]$ y $\beta_{e'}$ es cierto en un intervalo $[t_2, t_3]$ contiguo por la derecha (futuro) a $[t_1, t_2]$, donde $t < t_1 < t_2 < t_3$.

  La definición en LNint-e es: $ab^+(\alpha_e, \beta_{e'}) \stackrel{def}{=} F \uparrow\alpha_e \wedge \downarrow\alpha_e \approx^+ \uparrow\beta_{e'}$

Además de esto, en LNint-e se puede definir cualquier relación entre intervalos y puntos, entre las que podemos enumerar las siguientes: expresiones sobre el intervalo actual, expresiones de enunciados hereditarios, expresiones de puntos e intervalos, expresiones de puntos y eventos, expresiones de puntos e intervalos hereditarios y expresiones mixtas. Un estudio en profundidad de todas estas expresiones puede encontrarse en [27].

5. Introducción de forma normal

En esta sección introducimos una forma normal negativa para LNint-e$^+$. En su diseño tenemos en cuenta que estamos ante una lógica que presenta una dificultad considerable. De hecho, no conocemos en la bibliografía ningún precedente que siga este enfoque, a pesar de las multiples referencias sobre su necesidad.

Dos son las características que deseamos para la forma normal buscada:

1. Que evite cuanto trabajo sea posible al demostrador posterior.
2. Que tenga el menor número posible de conectivas temporales binarias, que introducen un elevado coste de manipulación a cualquier demostrador[10].

Hemos de destacar que emplearemos como conectivas binarias tanto $\preccurlyeq$ (primitiva en la construcción de LNint-e$^+$) como las conectivas definidas $\prec$, $\sqsubseteq$ y $\sqsubset$. Descartamos el uso de la conectiva $\approx^+$ ya que, al igual que ocurre con la conectiva booleana $\leftrightarrow$, representa una complejidad implícita que requiere un estudio específico.

Comenzamos con algunos resultados sobre m^+.

6. Resultados sobre m^+

En esta sección se incluyen algunos resultados de utilidad para el desarrollo de este trabajo. La mayoría de ellos se refieren al comportamiento de m^+ (el concepto básico en la semántica topológica de nuestra familia de lógicas) cuando se aplica a expresiones construidas con conectivas temporales, ya sean monarias o binarias.

El siguiente resultado es de comprobación inmediata.

Lema 6.1 *Sean A, B fbfs de LN$^+$, $n \in \mathbb{N}$, $t \in \mathbb{Z}$ y $\oplus^n l_p = \oplus\ .^n.\ \oplus l_p$ si $n > 0$ y $\oplus^0 l_p = l_p$. Entonces, se tiene:*

1. $m^+_{t\ (A\vee B)} = \min\{m^+_{tA}, m^+_{tB}\}$
2. $m^+_{t\ (A\wedge B)} \geq \max\{m^+_{tA}, m^+_{tB}\}$
3. $m^+_{t\ \oplus^n A} = m^+_{(t+1)\oplus^{(n-1)}A} - 1$, *si* $n > 1$
4. $m^+_{t\ F\oplus^n A} = \begin{cases} t+1 & si\ \ m^+_{(t+n+1)A} < \infty \\ +\infty & en\ otro\ caso \end{cases}$
5. $\begin{cases} m^+_{t\ G\oplus^n A} = t+1 \\ \quad si\ \ t+n+1 \in h(GA) \\ m^+_{t\ G\oplus^n A} = m^+_{(t+1)G\oplus^n A} \\ \quad si\ \ t+n+1 \in h(FGA \wedge F\neg A) \\ m^+_{t\ G\oplus^n A} = +\infty \\ \quad en\ otro\ caso \end{cases}$
6. $m^+_{t\ GFA} = \begin{cases} t+1 & si\ \ t \in h(GFA) \\ +\infty & en\ otro\ caso \end{cases}$
7. $m^+_{t\ FGA} = \begin{cases} t+1 & si\ t \in h(FGA) \\ +\infty & en\ otro\ caso \end{cases}$

7. Elección de literales

Los implicantes e implicados son ampliamente usados en varias áreas de la Inteligencia Artificial. Por ejemplo, se utilizan para crear modelos formales de sistemas de mantenimiento de verdad (TMSs) y sistemas de mantenimiento de verdad

[10] Los resultados presentados en [6, 5] permiten realizar un tratamiento completo de los literales monarios, por lo que disponer de conectivas temporales monarias en lugar de binarias aumenta la potencia de las transformaciones de reducción.

basados en asunción (ATMSs) [28]. Análogamente, son útiles para circunscripción [15], diagnosis basada en modelos [8], abducción [22, 24] y bases de datos relacionales [20].

La información contenida en los implicantes e implicados temporales puede ser gestionada mediante literales temporales definidos para cada lógica concreta [4]. La información contenida en estos literales se puede emplear para diseñar transformaciones de fórmulas temporales capaces de incrementar la potencia de las técnicas de deducción automática. En [6, 5] se realiza un estudio exhaustivo de los literales temporales monarios que permite asegurar un tratamiento completo y eficiente de los implicantes e implicados de las fórmulas de una lógica temporal sobre tiempo lineal discreto.

En nuestro proceso de normalización, y para conseguir los objetivos citados anteriormente, será crucial la elección del concepto de literal para nuestra lógica LNint-e$^+$.

7.1. Literales temporales con conectivas monarias

Definición 7.1 *Para cualquier átomo ξ, $\xi \in \Omega_{ins}^{ev\text{-}e}$, llamaremos* **literales en** ξ *a las fórmulas ξ y $\neg\xi$. En adelante, l_ξ denotará un literal en ξ y $\Omega_{ins}^{ev\text{-}e\,\pm}$ denotará el conjunto de los literales en ξ.*

Además, diferenciamos los siguientes tipos de literales en ξ:

- *Si $p \in \Omega_{ins}$, entonces l_p es un* **literal clásico en** p.
- *Si $\underline{t} \in \underline{\mathbb{Z}}$, entonces $l_{\underline{t}}$ es un* **literal de fecha en** t.
- *Si $\alpha_e \in EjecEventos$ y $\xi \in \{\uparrow\alpha_e, \overrightarrow{\alpha_e}, \downarrow\alpha_e\}$, entonces l_ξ es un* **literal de ejecución en** α_e.
- *Si $\alpha \in \Omega_{ev}$ y $\xi \in \{\uparrow\alpha_*, \overrightarrow{\alpha_*}, \downarrow\alpha_*\}$, entonces l_ξ es un* **literal de ejecución anónima en** ξ.

Notación:

En adelante denotamos por Ω_{lim}^+ al conjunto de los literales positivos de fecha o de ejecución, es decir,

$$\begin{aligned}\Omega_{lim}^+ = \; & \{t \in \mathbb{Z}\} \cup \\ & \{\uparrow\alpha_e, \overrightarrow{\alpha_e}, \downarrow\alpha_e \mid \alpha_e \in EjecEventos\}\end{aligned}$$

Obsérvese que los elementos de Ω_{lim}^+ son ciertos en un número finito [11] de instantes.

Además, denotamos por Ξ^{mon} al conjunto formado por las constantes y las fórmulas de LNint-e$^+$ que sólo contienen conectivas monarias, es decir,

$$\begin{aligned}\Xi^{mon} \; = \; & \{\top, \bot\} \cup \\ & \{\gamma_1 \ldots \gamma_k \, l_\xi \mid \gamma_i \in \{F, G, \oplus, \neg\} \text{ para} \\ & \text{todo } 1 \leq i \leq k, \; l_\xi \in \Omega_{ins}^{ev-e\,\pm}\}\end{aligned}$$

Definición 7.2 *Sea $A \in \Xi^{mon}$. Definimos el* opuesto *de A, denotado por $\overline{A}$, como sigue:*

- $\overline{\bot} = \top$ *y* $\overline{\top} = \bot$,
- $\overline{l_\xi} = \begin{cases} \neg\xi & si\ l_\xi = \xi \\ \xi & si\ l_\xi = \neg\xi \end{cases}$
- *Si $A = \gamma_1 \ldots \gamma_k \, l_\xi$, con $\gamma_i \in \{F, G, \oplus\}$ para todo $1 \leq i \leq k$, entonces:*

$$\overline{A} = \overline{\gamma_1} \ldots \overline{\gamma_k}\,\overline{l_\xi}$$

donde $\overline{\oplus} = \oplus$, $\overline{G} = F$ y $\overline{F} = G$.

Consideremos la relación de equivalencia semántica en LNint-e$^+$, que hemos denotado $\equiv$, y a partir de ella construimos el conjunto cociente $\Xi^{mon}/_\equiv$. Los lemas que se presentan a continuación permiten seleccionar un representante canónico para cada clase de equivalencia.

Lema 7.3 *Toda fórmula en Ξ^{mon} es equivalente semánticamente a una fórmula en el conjunto:*

$$\begin{aligned}&\{\top, \bot\} \cup \{FG\, l_\xi, GF\, l_\xi\} \cup \\ &\{\oplus^n l_\xi, F \oplus^n l_\xi, G \oplus^n l_\xi \mid n \in \mathbb{N}\}\end{aligned}$$

$$\text{donde } \oplus^n l_\xi = \begin{cases} \oplus \,.^n.\, \oplus l_\xi & si\ n > 0 \\ l_\xi & si\ n = 0 \end{cases}$$

Demostración:

Es suficiente considerar las siguientes leyes de equivalencia:

[11] Con el subíndice *lim* denotamos literales de apariciones *limitadas*.

(i) $\neg\neg A \equiv A; \quad \neg \oplus A \equiv \oplus\neg A$
$\neg FA \equiv G\neg A; \quad \neg GA \equiv F\neg A.$

(ii) $FFA \equiv F \oplus A; \quad GGA \equiv G \oplus A.$

(iii) *Si* $\gamma \in \{F, G\}$ *entonces* $\oplus\gamma A \equiv \gamma \oplus A.$

(iv) *Si* $\gamma \in \{FG, GF\}$ *entonces* $\gamma \oplus A \equiv \gamma A$, $F\gamma A \equiv \gamma A$ *y* $G\gamma A \equiv \gamma A.$

■

Es posible construir un algoritmo de coste lineal, denotado ***Canónico****, que proporciona el representante canónico de cada elemento de* $\Xi_{mon}/_{\equiv}$.

Definición 7.4 *Dado un literal en* ξ, $l_\xi \in \Omega_{ins}^{ev-e\pm}$, *el conjunto de* **literales temporales monarios en** ξ, *denotado* $\mathcal{L}it^{mon}(l_\xi)$, *es el conjunto definido de la siguiente manera:*

1. *Si* l_ξ *es un literal clásico o un literal de ejecución anónima, entonces*
$$\mathcal{L}it^{mon}(l_\xi) = \{\top, \bot\} \cup \{FG\, l_\xi, GF\, l_\xi\} \cup \{\oplus^n l_\xi, F \oplus^n l_\xi, G \oplus^n l_\xi \mid n \in \mathbb{N}\}$$
2. *Si* l_ξ *es un literal de fecha o un literal de ejecución, entonces*
$$\mathcal{L}it^{mon}(l_\xi) = \{\top, \bot\} \cup \{\oplus^n l_\xi, F \oplus^n l_\xi \mid n \in \mathbb{N}\}$$

El siguiente lema nos confirma la adecuación de la definición anterior.

Lema 7.5 *Para todo* $\alpha_e \in EjecEventos$ *y para todo* $m \in \mathbb{Z}$ *se tiene que si* $\gamma \in \{G, FG, GF\}$, *entonces:*

$$\gamma \uparrow\alpha_e \equiv \gamma \overrightarrow{\alpha_e} \equiv \gamma \downarrow\alpha_e \equiv \gamma \underline{m} \equiv \bot$$

Demostración:

Es consecuencia inmediata de la semántica de la conectiva G y de la unicidad de las fechas y ejecuciones de eventos de LNint-e (véase el apartado 3). ■

Definición 7.6 *Los elementos del conjunto* $\mathcal{L}it_{mon} = \bigcup_{l_\xi \in \Omega_{ins}^{ev-e\pm}} \mathcal{L}it_{mon}(l_\xi)$ *serán denominados* **literales temporales monarios**.

Considerando esto, cada literal temporal monario (no constante) expresa una de las siguientes aserciones:

- **Aserciones de punto:**
  - Ejecutable en un punto específico: $\oplus^n l_\xi$
  - Ejecutable en un punto no específico: $F \oplus^n l_\xi$
- **Aserciones de intervalo**:
  - Ejecutable en un punto específico: $G \oplus^n l_\xi$
  - Ejecutable en un punto no específico: FGl_ξ
- **Aserciones de secuencia infinita**: GFl_ξ.

Nos podemos preguntar que ocurre con las expresiones que especifican la ejecución de secuencias finitas. Tales expresiones, de interés para aplicaciones, vienen dadas por la conectiva definida que presentamos a continuación.

Definición 7.7 *Sean* $A_1, \ldots, A_n$ *expresiones booleanas de LNint-e*$^+$ *(es decir, fbfs que no tienen conectivas temporales) y* $m_1, \ldots, m_n \in \mathbb{N}$:

$$\begin{array}{lcl} F[A_1] & \stackrel{def}{=} & FA_1 \ y \\ F[A_1, \ldots, A_n] & \stackrel{def}{=} & F(A_1 \wedge F[A_2, \ldots, A_n]) \\ & & si \ n > 1 \end{array}$$

En particular, si $A_1 = \ldots = A_n = A$, *denotaremos* $F[A, \overset{n}{\ldots}, A]$ *como* $F^{[n]}A$. *Así,* $F[A_1, \ldots, A_n]$ *especifica la ejecución en el futuro de la secuencia* $A_1, \ldots, A_n$, *sin especificar los instantes de ejecución. Dualmente, definimos:*

$$\begin{array}{lcl} G[A_1] & \stackrel{def}{=} & GA_1 \quad y \\ G[A_1, \ldots, A_n] & \stackrel{def}{=} & G(A_1 \vee G[A_2, \ldots, A_n]) \\ & & si \quad n > 1 \end{array}$$

Si $A_1 = \ldots = A_n = A$, $G[A, \overset{n}{\ldots}, A]$ *es denotado por* $G^{[n]}A$.

La siguiente proposición establece el correcto comportamiento de las conectivas definidas presentadas en la definición anterior con respecto a las conectivas temporales monarias.

Proposición 7.8 *Sean* $m_1, \ldots, m_n \in \mathbb{N}$ *y sean* $A_1, \ldots, A_n$ *expresiones booleanas de LNint-e*$^+$, *entonces*

1. $\oplus F[A_1,\ldots,A_n] \equiv F[\oplus A_1,\ldots,\oplus A_n]$ *y* $\oplus F^{[n]}A \equiv F^{[n]} \oplus A$.
2. $F\,F[A_1,\ldots,A_n] \equiv F[\oplus A_1,\ldots,\oplus A_n]$ *y* $F\,F^{[n]}A \equiv F^{[n]} \oplus A$.
3. $G\,F[A_1,\ldots,A_n] \equiv \bigwedge_{i=1}^{i=n} GFA_i$ *y* $G\,F^{[n]}A \equiv GF\,A$

Los resultados duales son también ciertos.

Definición 7.9 *$\mathcal{L}it^*_{mon}$ es la extensión de $\mathcal{L}it_{mon}$ definida como sigue:*

$\mathcal{L}it^*_{mon} = \mathcal{L}it_{mon} \cup$

$\{F[l_1,\ldots,l_n], G[l_1,\ldots,l_n] \mid l_1,\ldots,l_n \in \mathcal{L}it_{mon}\}$

En adelante, y con la sola intención de simplificar la complejidad sintáctica cuando se consideran conectivas temporales futuras puras[12], introducimos la conectiva "ayer", denotada $\ominus$ y cuya semántica es $h(\ominus A) = \{t \in \mathbb{Z} \mid t-1 \in h(A)\}$. Consecuentemente, $\oplus \ominus A \equiv \ominus \oplus A \equiv A$; $\ominus FA \equiv F \ominus A \equiv A \vee FA$ y $\ominus GA \equiv G \ominus A \equiv A \wedge GA$. [13]

7.2. Literales temporales con conectivas binarias

Antes de definir el conjunto de literales de LNint-e$^+$, necesitamos algunos resultados que establecen las condiciones para reemplazar conectivas temporales binarias por conectivas temporales monarias.

Definición 7.10 *En LNint-e$^+$ definimos una relación binaria, $\trianglelefteq$, del siguiente modo: sean A, B fbfs de LNint-e$^+$*

$$A \trianglelefteq B \text{ si y sólo si } \models A \rightarrow B$$

Proposición 7.11 *Sean A y B fbfs de LNint-e$^+$, entonces se satisfacen las siguientes equivalencias:*

1. $A \preccurlyeq A \equiv F\,A; \quad A \prec A \equiv \bot$
   $A \sqsubseteq A \equiv \top; \quad A \sqsubset A \equiv G\,\overline{A}$.
2. *Si $A \trianglelefteq B$, entonces: $B \sqsubseteq A \equiv \top$; $A \prec B \equiv \bot$; $B \preccurlyeq A \equiv FB$ y $A \sqsubset B \equiv G\overline{B}$.*
3. *Si $\overline{A} \trianglelefteq B$, entonces: $A \preccurlyeq B \equiv A \sqsubseteq B \equiv \oplus A$ y $B \sqsubset A \equiv B \prec A \equiv \oplus\neg A$.*

Demostración:

La demostración del ítem 1 es inmediata a partir de la semántica de las conectivas. Demostraremos algunas de las equivalencias de los ítemes 2 y 3. El resto de los ítemes se demuestra analogamente.

Si $A \trianglelefteq B$, por definición se tiene que $\models A \rightarrow B$ y entonces $h(A) \subseteq h(B)$. Por tanto, para todo $t \in \mathbb{Z}$:

$m^+_{tA} =$
$min((t,+\infty) \cap h(A)) \geq min((t,+\infty) \cap h(B)) =$
m^+_{tB}

Entonces, la definición de $\sqsubseteq$, asegura que $B \sqsubseteq A \equiv \top$.

Si $\overline{A} \trianglelefteq B$, entonces se tiene que $h(\overline{A}) \subseteq h(B)$. Sea $t \in h(A \preccurlyeq B)$, entonces, $m^+_{tA} \leq m^+_{tB}$ y $m^+_{tA} < +\infty$. Por definición de m^+_{tA}, se tiene que $m^+_{tA} \geq t+1$.

Supongamos que $m^+_{tA} > t+1$. Entonces, se obtiene que $t+1 \in h(\neg A)$ y consecuentemente $t+1 \in h(B)$. Esta situación asegura $m^+_{tB} < m^+_{tA}$, lo que contradice la hipótesis. Por tanto, concluimos que $m^+_{tA} = t+1$, es decir, $t \in h(\oplus A)$.

Por otra parte, demostrar que $h(\oplus A) \subseteq h(A \preccurlyeq B)$ es trivial y finalmente se tiene que $A \preccurlyeq B \equiv \oplus A$.

∎

Proposición 7.12 *Sean A y B fbfs de LNint-e$^+$, entonces*

(a)

$$\begin{array}{lcl} FA \preccurlyeq B & \equiv & F \oplus A \\ FA \prec B & \equiv & F \oplus A \wedge \oplus\neg B \\ FA \sqsubseteq B & \equiv & F \oplus A \vee G\neg B \\ FA \sqsubset B & \equiv & (F \oplus A \wedge \oplus\neg B) \vee G\neg B \end{array}$$

[12] En este punto nos apartamos de la bibliografía, que usa conectivas temporales de futuro incluyendo el presente. El motivo de nuestra opción es que nuestra aproximación a las aplicaciones nos hace inclinarnos por el paradigma *Pasado Declarativo y Futuro Imperativo* [14], que contempla cada fórmula como una combinación booleana de fórmulas separadas, esto es, fórmulas de pasado puro, presente puro y futuro puro.

[13] El uso de $\ominus$ no produce ningún conflicto con el hecho de estar contemplando el fragmento de futuro de LNint-e, ya que, como veremos más adelante, usaremos $\ominus$ exclusivamente en literales de futuro puro.

(b)

$$\begin{aligned} A \prec FB &\equiv FA \wedge G \oplus \neg B \\ A \sqsubseteq FB &\equiv \oplus A \vee G \oplus \neg B \\ A \sqsubset FB &\equiv G \oplus \neg B \\ A \preccurlyeq FB &\equiv (G \oplus \neg B \vee \oplus A) \wedge FA \end{aligned}$$

(c)

$$\begin{aligned} GA \prec B &\equiv F(GA \wedge \neg B) \\ GA \sqsubseteq B &\equiv G[\neg B, A] \\ GA \sqsubset B &\equiv \oplus \neg B \wedge G[\neg B, \ominus A] \\ GA \preccurlyeq B &\equiv FGA \wedge G[\neg B, A] \end{aligned}$$

(d)

$$\begin{aligned} A \prec GB &\equiv F[A, \neg B] \\ A \sqsubseteq GB &\equiv G(F\neg B \vee A) \\ A \sqsubset GB &\equiv GF\neg B \vee F[A, \neg B] \\ A \preccurlyeq GB &\equiv \oplus A \vee F[A, \ominus \neg B] \end{aligned}$$

(e) Sea $\gamma \in \{FG, GF\}$, entonces:

$$\begin{aligned} \gamma A \preccurlyeq B &\equiv \gamma A \\ \gamma A \prec B &\equiv \gamma A \wedge \oplus \neg B \\ \gamma A \sqsubseteq B &\equiv \gamma A \vee G\neg B \\ \gamma A \sqsubset B &\equiv (\gamma A \wedge \oplus \neg B) \vee G\neg B \end{aligned}$$

(f) Sea $\gamma \in \{FG, GF\}$, entonces:

$$\begin{aligned} A \prec \gamma B &\equiv \bar{\gamma}\neg B \wedge FA \\ A \sqsubseteq \gamma B &\equiv \bar{\gamma}\neg B \vee \oplus A \\ A \sqsubset \gamma B &\equiv \bar{\gamma}\neg B \\ A \preccurlyeq \gamma B &\equiv (\bar{\gamma}\neg B \vee \oplus A) \wedge FA \end{aligned}$$

DEMOSTRACIÓN:

Demostramos algunos de los ítemes. El resto de las demostraciones son similares:

(a) Demostramos cada equivalencia por separado:

- Para toda interpretación temporal h, se tiene que $t \in h(FA \preccurlyeq B)$ si y sólo si $m^+_{t\,FA} \leq m^+_{t\,B}$ y $m^+_{t\,FA} < +\infty$. Por el lema 6.1 esta situación se satisface si y sólo si $m^+_{t\,FA} = t+1$, es decir, $t \in h(F \oplus A)$ y, por tanto, $FA \preccurlyeq B \equiv F \oplus A$.
- Puesto que $FA \sqsubseteq B \equiv A \preccurlyeq B \vee G\neg B$, se tiene que $FA \sqsubseteq B \equiv F \oplus A \vee G\neg B$.
- Para toda interpretación temporal h, se tiene que $t \in h(FA \prec B)$ si y sólo si $m^+_{t\,FA} < m^+_{t\,B}$. Por el lema 6.1, esto se satisface si y sólo si $m^+_{t\,FA} = t+1$ y $t \in h(\oplus \neg A)$. Por tanto, $FA \prec B \equiv F \oplus A \wedge \oplus \neg B$.
- De $FA \sqsubset B \equiv A \prec B \vee G\neg B$, deducimos que $FA \sqsubset B \equiv (F \oplus A \wedge \oplus \neg B) \vee G\neg B$.

(e) Para toda interpretación temporal h, se tiene que $t \in h(FGA \preccurlyeq B)$ si y sólo si $m^+_{t\,FGA} \leq m^+_{t\,B}$ y $m^+_{t\,FGA} < +\infty$. Por el lema 6.1, esta situación se satisface si y sólo si $m^+_{t\,FGA} = t+1$, es decir, $t \in h(FGA)$. Por tanto, concluimos que $\gamma A \preccurlyeq B \equiv \gamma A$.

El resto de las demostraciones se puede construir fácilmente siguiendo el ítem (a).

■

Ahora tenemos todos los elementos necesarios para abordar la definición de literal en LNint-e$^+$. Un primer intento nos llevaría a considerar un conjunto de literales Γ, definido de la siguiente forma:

$$\begin{aligned} \Gamma = {} & \mathcal{L}it_{mon} \cup \\ & \{l_1 \odot l_2 \mid l_1, l_2 \in \mathcal{L}it_{mon}, \odot \in \{\preccurlyeq, \prec, \sqsubseteq, \sqsubset\}\} \end{aligned}$$

Sin embargo, podemos asegurar el siguiente resultado:

Teorema 7.13 *Sea A una fbf en Γ, entonces existe una fórmula equivalente a A en el siguiente conjunto:*

$$\begin{aligned} & \mathcal{B}ool(\mathcal{L}it^*_{mon}) \cup \\ & \{\oplus^n l_{p_1} \odot \oplus^m l_{p_2} \mid p_1, p_2 \in \Omega^{\pm}, \odot \in \{\preccurlyeq, \prec, \sqsubseteq, \sqsubset\}\} \end{aligned}$$

donde $\mathcal{B}ool(\mathcal{L}it^*_{mon})$ denota el conjunto de todas las expresiones boolenas construidas con elementos de $\mathcal{L}it^*_{mon}$.

DEMOSTRACIÓN: La demostración es una consecuencia directa de las proposiciones 7.11 y 7.12 aplicadas a todos los elementos del conjunto $\{l_1 \odot l_2 \mid l_1, l_2 \in \mathcal{L}it_{mon}, \odot \in \{\preccurlyeq, \prec, \sqsubseteq, \sqsubset\}\}$.

■

Como en el caso de $\Xi_{mon}/_{\equiv}$, podemos asegurar que existe un algoritmo de coste lineal que proporciona el representante canónico de cada elemento de $\Gamma/_{\equiv}$. Denotaremos también este algoritmo como **Canonic**.

Definición 7.14 *Un* **literal temporal binario** *de LNint-e$^+$ es un elemento del siguiente conjunto:*

$$\begin{aligned} & \mathcal{L}it_{bin} = \\ & \{\oplus^n l_{p_1} \odot \oplus^m l_{p_2} \mid p_1, p_2 \in \Omega^{\pm}, \odot \in \{\preccurlyeq, \prec, \sqsubseteq, \sqsubset\}\} \end{aligned}$$

Finalmente, tenemos la deseada definición de literal.

Definición 7.15 *Llamaremos* **literal temporal** *de LNint-e$^+$ a los elementos del conjunto:* [14]

$$\mathcal{L}it = \mathcal{B}ool(\mathcal{L}it^*_{mon}) \cup \mathcal{L}it_{bin}$$

Puesto que trabajaremos únicamente con literales temporales, en adelante omitiremos el adjetivo *temporal*.

8. Transformaciones de simplificación

En esta sección, una vez introducido el concepto de literal, definiremos cada una de las transformaciones en LNint-e$^+$ que toman parte en el proceso de normalización. Todas las transformaciones que vamos a definir son transformaciones de equivalencia, esto es, transforman una fbf en otra equivalente con las propiedades deseadas.

I) **Transmisión de negaciones a los átomos**, denotada **TNeg**: aplicando las *leyes de negación de de Morgan*, la *interdefinición* de F y G y las leyes $\neg(A \preccurlyeq B) \equiv B \sqsubset A$ y $\neg(A \prec B) \equiv B \sqsubseteq A$.

II) **Eliminación de constantes** denotada **EConst**: por medio de las leyes $0-1$ de la lógica clásica y las siguientes leyes (y también sus duales):[15]

$$\begin{array}{l} \oplus\top \equiv F\top \equiv G\top \equiv \top \\ \top \preccurlyeq A \equiv \top \sqsubseteq A \equiv \top \\ A \preccurlyeq \top \equiv A \sqsubseteq \top \equiv \oplus A \\ A \preccurlyeq \bot \equiv A \prec \bot \equiv FA \\ \bot \preccurlyeq A \equiv \bot \prec A \equiv \bot \end{array}$$

Proposición 8.1 *La transformación* **EConst** *es una transformación de equivalencia.*

Demostración: Demostramos $\bot \sqsubset A \equiv G\neg A$ como un ejemplo ilustrativo. El resto de los casos se demuestra de manera análoga:

$$\begin{array}{l} h(\bot \sqsubset A) = \\ \{t \in \mathbb{Z} \mid m^+_{t\bot} < m^+_{tA} \text{ or } m^+_{tA} = +\infty\} = \\ \{t \in \mathbb{Z} \mid m^+_{tA} = +\infty\} = \\ h(G\neg A) \end{array}$$

■

III) **Introducción de constantes**, denotada **IConst**, por medio de:

- Las leyes de la proposición 7.11:
  $A \prec A \equiv \bot$, $A \sqsubseteq A \equiv \top$; si $A \trianglelefteq B$ entonces $B \sqsubseteq A \equiv \top$ y $A \prec B \equiv \bot$.
- $l \wedge \bar{l} \equiv \bot$ y $l \vee \bar{l} \equiv \top$, para todo $l \in \mathcal{L}it$.
- $\gamma \uparrow \alpha_e \equiv \gamma\overrightarrow{\alpha_e} \equiv \gamma \downarrow \alpha_e \equiv \gamma\underline{m} \equiv \bot$ para todo $\alpha_e \in EjecEventos$, todo $m \in \mathbb{Z}$ y $\gamma \in \{G, FG, GF\}$ (estas leyes fueron usadas en la definición de $\mathcal{L}it$).

Y además disponemos de las siguientes leyes:

Proposición 8.2

1. *Si $\gamma \in \{\oplus, F\}$ y $\alpha_e \in EjecEventos$, entonces:*
$$\uparrow\alpha_e \wedge \gamma \uparrow\alpha_e \equiv \downarrow\alpha_e \wedge \gamma \downarrow\alpha_e \equiv \bot$$
2. *Si $\gamma \in \{\oplus, F\}$ y $m \in \mathbb{Z}$, entonces $\underline{m} \wedge \gamma \underline{m} \equiv \bot$.*
3. *Si $\odot \in \{\preccurlyeq, \prec, \sqsubseteq, \sqsubset\}$ y $\alpha_e \in EjecEventos$, entonces:*
$$\uparrow\alpha_e \odot \downarrow\alpha_e \equiv \uparrow\alpha_e \odot \overrightarrow{\alpha_e} \equiv \overrightarrow{\alpha_e} \odot \downarrow\alpha_e \equiv \top$$
$$\downarrow\alpha_e \odot \uparrow\alpha_e \equiv \downarrow\alpha_e \odot \overrightarrow{\alpha_e} \equiv \overrightarrow{\alpha_e} \odot \uparrow\alpha_e \equiv \bot$$
4. *Si $\gamma \in \{G, GF, FG\}$, $l \in \Omega^+_{lim}$ y A es una* fbf *de LNint-e$^+$, entonces*
$$\gamma(l \prec A) \equiv \gamma(l \preccurlyeq A) \equiv \bot$$

Demostración:

Demostramos, como ejemplo, una de las equivalencias recogidas en el ítem 3, en particular, $FG(l \prec A)) \equiv \bot$.

Supongamos que existe $t \in h(FG(l \prec A))$. Esto es cierto si y sólo si existe $t' > t$ tal que

[14] Obsérvese que $\mathcal{L}it$ es el conjunto identificado en el lema 7.13.

[15] El interés de las leyes relativas a la eliminación de constantes se debe al hecho de que, aunque estas no aparecen usualmente en las especificaciones y, por tanto, no forman parte de las fórmulas de entrada de los demostradores, pueden ser introducidas por otras transformaciones del propio proceso de normalización.

$t' \in h(G(l \prec A))$, y, por tanto, para todo $t'' > t'$ se verifica que $t'' \in h((l \prec A))$. Por la semántica de la conectiva $\prec$, esto implica infinitas apariciones de l; pero esto contradice la restricción de unicidad de fechas y ejecuciones en LNint-e$^+$ (véase el apartado 3).

■

Corolario 8.3 *Sea $l \in \Omega^+_{lim}$ un literal de apariciones limitadas. Entonces se verifica que*

$$\bot \sqsubseteq \neg l \quad \equiv \quad \bot \sqsubset \neg l \quad \equiv \bot$$

DEMOSTRACIÓN:

Directa, a partir de las proposiciones 8.1 y 8.2, ya que

$$\begin{array}{lllllll} \bot \sqsubseteq \neg l & \equiv & G\neg\neg l & \equiv & Gl & \equiv & \bot \\ \bot \sqsubset \neg l & \equiv & G\neg\neg l & \equiv & Gl & \equiv & \bot \end{array}$$

■

IV) **Reducción del ámbito de las conectivas temporales por medio de distributividad**, denotada **Dist**. El objetivo de esta transformación es "bajar" las conectivas temporales a los átomos y, consecuentemente, reducir su complejidad tanto para recuperar su información como para el procesamiento posterior. Para ello, usaremos la leyes de distributividad de G con respecto a $\wedge$ y de F con respecto a $\vee$, junto con las siguientes leyes:

Sean A, B, A_i, B_i fbfs de LNint-e$^+$, entonces:

1. $\oplus(A \odot B) \equiv \oplus A \odot \oplus B$, donde $\odot \in \{\wedge, \vee, \preccurlyeq, \prec, \sqsubseteq, \sqsubset\}$.
2. $F(\bigwedge_{i=1}^{i=n} A_i) \equiv \bigwedge_{i=1}^{i=n} FA_i$ y
   $G(\bigvee_{i=1}^{i=n} A_i) \equiv \bigvee_{i=1}^{i=n} GA_i$
   si se cumple una de las siguientes condiciones:
   (i) $A_i = FB_i$ para todo $i \in \{1, \ldots, n\}$.
   (ii) $A_i = GB_i$ para todo $i \in \{1, \ldots, n\}$.
3. Sean $I = \{1, \ldots, n\}$, $J = \{i \in I \mid A_i = \gamma_i B_i$ y $\gamma_i \in \{FG, GF\}\}$, entonces se tiene:
   $F(\bigwedge_{i\in I} A_i) \equiv (\bigwedge_{i\in J} A_i) \wedge F(\bigwedge_{i\in I\setminus J} A_i)$
   $G(\bigvee_{i\in I} A_i) \equiv (\bigvee_{i\in J} A_i) \vee G(\bigvee_{i\in I\setminus J} A_i)$.
4. Sean $I = \{1, \ldots, n\}$, $J = \{i \in I \mid A_i = \gamma_i B_i$, $\gamma_i \in \{FG, GF\}\}$ y sea $\odot \in \{\preccurlyeq, \prec\}$, entonces se tiene:
   $(\bigwedge_{i\in I} A_i) \preccurlyeq B \equiv (\bigwedge_{i\in J} A_i) \wedge (\bigwedge_{i\in I\setminus J} A_i) \preccurlyeq B$
   $(\bigwedge_{i\in I} A_i) \prec B \equiv (\bigwedge_{i\in J} A_i) \wedge (\bigwedge_{i\in I\setminus J} A_i) \prec B$

Proposición 8.4 *La transformación* **Dist** *es una transformación de equivalencia.*

DEMOSTRACIÓN: Demostramos el ítem (a) para la conectiva $\preccurlyeq$ y el ítem (d) para la conectiva $\prec$. El resto de los casos se demuestran análogamente.

(a) Se tiene que $t \in h(\oplus(A \preccurlyeq B))$ si y sólo si $t+1 \in h(A \preccurlyeq B)$, es decir, (por definición de $\preccurlyeq$)

$m^+_{(t+1)A} < \infty$ y $m^+_{(t+1)A} \leq m^+_{(t+1)B}$ si y sólo si (por el lema 6.1)

$m^+_{t\oplus A} < \infty$ y $m^+_{t\oplus A} \leq m^+_{t\oplus B}$ si y sólo si $t \in h(\oplus A \preccurlyeq \oplus B)$.

(d) Puesto que $(\bigwedge_{i\in I} A_i) \prec B \equiv (\bigwedge_{i\in J} A_i \prec B) \wedge (\bigwedge_{i\in I\setminus J} A_i) \prec B$, es suficiente demostrar que $(\bigwedge_{i\in J} A_i) \prec B \equiv \bigwedge_{i\in J} A_i$. Para toda interpretación temporal h, se tiene que $t \in h((\bigwedge_{i\in J} A_i) \prec B)$ si y sólo si $m^+_{t\,\bigwedge_{i\in J} A_i} < m^+_{t\,B}$ y esto es cierto si y sólo si $m^+_{t\,\bigwedge_{i\in J} A_i} < +\infty$.

Por la semántica de FG y GF se tiene que $h(\bigwedge_{i\in J} A_i) = \mathbb{Z}$ y, por ello, $t \in h(\bigwedge_{i\in J} A_i)$. En consecuencia

$$(\bigwedge_{i\in I} A_i) \prec B \equiv (\bigwedge_{i\in J} A_i) \wedge (\bigwedge_{i\in I\setminus J} A_i) \prec B$$

■

V) **Eliminación de conectivas temporales binarias**. Disponemos de dos transformaciones para dicha eliminación:

- La primera de ellas, denotada **EBin1**, aplica leyes que reducen fórmulas en las que el argumento de una conectiva temporal binaria está en el ámbito de una conectiva temporal monaria: para todo par de fórmulas A y B en LNint-e$^+$:

(a)

$$\begin{array}{lcl} G(A \prec B) & \equiv & GFA \wedge G \oplus \neg B \\ G(A \sqsubseteq B) & \equiv & G(\oplus \neg B \vee \oplus A) \\ G(A \sqsubset B) & \equiv & G \oplus \neg B \end{array}$$

(b)

$$\begin{array}{lcl} F(A \preccurlyeq B) & \equiv & F \oplus A \\ F(A \prec B) & \equiv & F \oplus (A \wedge \neg B) \\ F(A \sqsubseteq B) & \equiv & FG\neg B \vee F \oplus A \end{array}$$

(c)

$$\begin{array}{lcl} FG(A \prec B) & \equiv & GFA \wedge FG\neg B \\ FG(A \sqsubseteq B) & \equiv & FG(\neg B \vee A) \\ FG(A \sqsubset B) & \equiv & FG\neg B \end{array}$$

(d)

$$\begin{array}{lcl} GF(A \preccurlyeq B) & \equiv & GFA \\ GF(A \prec B) & \equiv & GF(A \wedge \neg B) \\ GF(A \sqsubseteq B) & \equiv & FG\neg B \vee GFA \end{array}$$

Proposición 8.5 *La transformación* **EBin1** *es una transformación de equivalencia.*

DEMOSTRACIÓN: Demostramos algunos de los ítemes. El resto de las demostraciones son similares:

(a) $t \in h(G(A \prec B))$ si y sólo si
para todo $x \geq t+1$, $m^+_{xA} < \infty$ y $m^+_{xA} < m^+_{xB}$, si y sólo si
para todo $x \geq t+1, x \in h(FA)$ y $(x, m^+_{xA}) \subseteq (x, m^+_{xB})$, si y sólo si
para todo $x \geq t+1$, $x \in h(FA)$ y se tienen los dos casos siguientes:
si $x+1 \in h(\neg A)$ entonces $x+1 \in h(\neg B)$
si $x+1 \in h(A)$ entonces $x+1 \in h(\neg B)$
si y sólo si
para todo $x \geq t+1$, $x \in h(FA)$ y $x \in h(\oplus \neg B)$ si y sólo si
$t \in h(GFA)$ y $t \in h(G \oplus \neg B)$ si y sólo si
$t \in h(GFA \wedge G \oplus \neg B)$

(b) Se deduce directamente de la negación del ítem anterior y de **TNeg**.

(c) Usando el ítem (b) se tiene que:
$FG(A \prec B) \equiv F(\, G(A \prec B)\,) \equiv F(\, GFA \wedge G \oplus \neg B\,) \stackrel{\textbf{Dist}}{\equiv}$
$FGFA \wedge FG \oplus \neg B \equiv GFA \wedge FG\neg B$

(d) Es una conclusión directa de la negación del ítem (c) y **TNeg**.

■

Corolario 8.6 *Sea A una* fbf *de LNint-e y* $l, l' \in \Omega^+_{lim}$*, entonces:*

$$\begin{array}{rcl} G(A \prec \neg l) \equiv G(l \sqsubseteq \neg l') \equiv G(A \sqsubset \neg l) & \equiv & \bot \\ FG(A \prec \neg l) \equiv FG(l \sqsubseteq \neg l') & \equiv & \\ FG(A \sqsubset \neg l) & \equiv & \bot \\ GF(A \prec \neg l) \equiv GF(l \sqsubseteq \neg l') & \equiv & \bot \end{array}$$

DEMOSTRACIÓN:

Directa a partir de las proposiciones 8.2 y 8.5. Demostramos algunas equivalencias como ejemplo:

$$\begin{array}{lcll} G(A \prec \neg l) & \equiv & GFA \wedge G \oplus \neg\neg l & \equiv \\ & & GFA \wedge \bot & \equiv \bot \\ G(A \sqsubset \neg l) & \equiv & G \oplus \neg\neg l & \equiv \bot \\ FG(A \sqsubset \neg l) & \equiv & FG \oplus \neg\neg l & \equiv \bot \end{array}$$

■

Obsérvese que otras posibles simplificaciones en expresiones del tipo $\gamma(l \odot B)$, con $\gamma \in \{F, G, FG, GF\}, l \in \Omega^+_{lim}, \odot \in \{\preccurlyeq, \prec\}$ ya están contempladas en el lema 8.2.

- La segunda transformación de eliminación de conectivas binarias, denotada **EBin2**, aplica aquellas leyes de la proposición 7.12 que no incrementan la complejidad de la fórmulas, esto es, las siguientes leyes:

(a) $FA \preccurlyeq B \equiv F \oplus A$
$FA \prec B \equiv F \oplus A \wedge \oplus \neg B$
$FA \sqsubseteq B \equiv F \oplus A \vee G\neg B$

(b) $A \prec FB \equiv FA \wedge G \oplus \neg B$
$A \sqsubseteq FB \equiv \oplus A \vee G \oplus \neg B$
$A \sqsubset FB \equiv G \oplus \neg B$

(c) $GA \prec B \equiv F(GA \wedge \neg B)$
$GA \sqsubseteq B \equiv G(\neg B \vee GA)$

(d) $A \prec GB \equiv F(A \wedge F\neg B)$
$A \sqsubseteq GB \equiv G(F\neg B \vee A))$

(e) Sea $\gamma \in \{FG, GF\}$, entonces:
$\gamma A \preccurlyeq B \equiv \gamma A$
$\gamma A \prec B \equiv \gamma A \wedge \oplus \neg B$
$\gamma A \sqsubseteq B \equiv \gamma A \vee G\neg B$

(f) Sea $\gamma \in \{FG, GF\}$, entonces:

$$A \prec \gamma B \equiv \bar{\gamma}\neg B \wedge FA$$

$$A \sqsubseteq \gamma B \equiv \bar{\gamma}\neg B \vee \oplus A$$

$$A \sqsubset \gamma B \equiv \bar{\gamma}\neg B$$

Corolario 8.7 *Sea A una* fbf *de LNint-e, $l, l' \in \Omega^+_{lim}$ y $\gamma \in \{FG, GF\}$, entonces:*

$$\begin{array}{lcl} A \prec F\neg l \equiv A \sqsubset F\neg l & \equiv & \bot \\ Gl \sqsubseteq \neg l' \equiv l \sqsubseteq G\neg l' & \equiv & \bot \\ \gamma l \preccurlyeq A \equiv \gamma l \prec A \equiv \gamma l \sqsubseteq \neg l' & \equiv & \bot \\ A \preccurlyeq \gamma\neg l \equiv A \sqsubset \gamma\neg l & \equiv & \bot \end{array}$$

Demostración:

Directa a partir de los lemas 7.12 y 8.2. Demostramos algunas equivalencias como ejemplos:

$$\begin{array}{rcl} A \prec F\neg l \equiv FA \wedge G \oplus \neg\neg l \equiv FA \wedge \bot & \equiv & \bot \\ A \sqsubset F\neg l \equiv G \oplus \neg\neg l & \equiv & \bot \end{array}$$

■

Ejemplo 8.1 *Veamos algunos ejemplos de aplicación de las transformaciones anteriores:*

1. $G(l \prec l) \overset{\textbf{IConst}}{\equiv} G(\bot) \equiv \bot$

2. $G(\downarrow\alpha_e \prec \uparrow\alpha_e) \overset{\textbf{8.2}}{\equiv} G(\bot) \overset{\textbf{EConst}}{\equiv} \bot$

3. $G(\neg(p \preccurlyeq B)) \overset{\textbf{TNeg}}{\equiv} G(B \sqsubset p) \overset{\textbf{EBin1}-\textbf{(a)}}{\equiv} G \oplus \neg p$

4. $F(p \vee FGq) \preccurlyeq (\neg \oplus r \sqsubset \oplus Fs) \overset{\textbf{EBin2}-\textbf{(a)}}{\equiv} F \oplus (p \vee FGq) \overset{\textbf{Dist}}{\equiv} F \oplus p \vee FGq$

5. $FG\neg A \preccurlyeq B \wedge G \oplus C \prec B \wedge (FD \wedge GE) \preccurlyeq B \overset{\textbf{EBin2}-\textbf{(c) y (e)}}{\equiv} FG\neg A \wedge F(G \oplus C \wedge \neg B) \wedge (FD \wedge GE) \preccurlyeq B$

6. $F \oplus [FG(B \prec C) \wedge GF(A \sqsubseteq B) \wedge \neg(D \preccurlyeq C)] \overset{\textbf{Dist}-\textbf{(1)}}{\equiv} F[FG(B \prec C) \wedge GF(A \sqsubseteq B) \wedge \neg(\oplus D \preccurlyeq \oplus C)] \overset{\textbf{Dist}-\textbf{(3).i}}{\equiv} FG(B \prec C) \wedge GF(A \sqsubseteq B) \wedge F\neg(\oplus D \preccurlyeq \oplus C) \overset{\textbf{TNeg}}{\equiv} FG(B \prec C) \wedge GF(A \sqsubseteq B) \wedge F(\oplus C \sqsubset \oplus D) \overset{\textbf{EBin1}-\textbf{(c) y (d)}}{\equiv} GFB \wedge FG\neg C \wedge (FG\neg B \vee GFA) \wedge F(\oplus C \sqsubset \oplus D)$

9. Forma normal negativa en LNint-e$^+$

Definición 9.1

Una fbf ϕ de LNint-e$^+$ está en forma normal negativa, denotada fnn_t, si se cumplen las siguientes condiciones:

1. *Pertenece a la clausura inductiva de $\mathcal{L}it$ libremente generada por las conectivas $\{\wedge, \vee, \oplus, F, G, FG, GF, \preccurlyeq, \prec, \sqsubseteq, \sqsubset\}$.*
2. *Es óptima en el sentido de que no contiene ninguna aparición de las conectivas $\preccurlyeq, \prec, \sqsubseteq, \sqsubset$ reducible a $\oplus, F, G, FG, GF$ sin incrementar la complejidad de la fórmula.*

Obsérvese que:

- La segunda condición asegura que no existe ninguna aserción de precedencia reducible a una expresión booleana construida con aserciones de punto, intervalo o secuencia (en el sentido del apartado 7). Nótese que esta transformación sólo se aplica cuando no se aumenta la complejidad de la fórmula.
- Las aserciones básicas son expresiones booleanas de punto (específico o no específico), intervalo (específico o no específico) o secuencia (finita o infinita) de literales de la lógica clásica proposicional.

Teorema 9.2

Para cada fbf ϕ de LNint-e$^+$ existe una fnn_t ψ tal que $\phi \equiv \psi$.

9.1. Ejemplos

A continuación mostramos dos ejemplos completos del proceso de obtención de la forma normal negativa temporal de LNint-e$^+$.

Ejemplo 9.1 *Supongamos que la fórmula de entrada es* $F(\neg((A \sqsubset \underline{m}) \prec C))$.

$$
\begin{array}{ll}
F(\neg((A \sqsubset \underline{m}) \prec C)) & \overset{\textbf{TNeg}}{\equiv} \\
F(C \sqsubseteq (A \sqsubset \underline{m})) & \overset{\textbf{EBin1}-\textbf{(b)}}{\equiv} \\
F \oplus C \vee FG\neg(A \sqsubset \underline{m}) & \overset{\textbf{TNeg}}{\equiv} \\
F \oplus C \vee FG(\underline{m} \preccurlyeq A) & \overset{\textbf{8,2}}{\equiv} \\
F \oplus C \vee \bot & \equiv \\
F \oplus C &
\end{array}
$$

Ejemplo 9.2 *Supongamos que la fórmula de entrada es*

$$G(FA \sqsubset FG(B \preccurlyeq \oplus C)) \wedge (GF\, D \prec \neg(FA \sqsubseteq B))$$

Analizaremos separadamente cada una de las subfórmulas de la conjunción. Por una parte tenemos:

$$
\begin{array}{ll}
G(FA \sqsubset FG(B \preccurlyeq \oplus C)) & \overset{\textbf{EBin1}-\textbf{(a)}}{\equiv} \\
G \oplus \neg FG(B \preccurlyeq \oplus C) & \overset{\textbf{Canónico}}{\equiv} \\
GF\neg(B \preccurlyeq \oplus C) & \overset{\textbf{TNeg}}{\equiv} \\
GF(\oplus C \sqsubset B) &
\end{array}
$$

La segunda subfórmula se procesa como sigue:

$$
\begin{array}{ll}
GF\, D \prec \neg(FA \sqsubseteq B) & \overset{\textbf{EBin2}-\textbf{(c)}}{\equiv} \\
GF\, D \wedge \oplus\neg\neg(FA \sqsubseteq B) & \overset{\textbf{Canónico}}{\equiv} \\
GF\, D \wedge \oplus(FA \sqsubseteq B) & \overset{\textbf{Dist}-\textbf{(1)}}{\equiv} \\
GF\, D \wedge \oplus FA \sqsubseteq \oplus B & \overset{\textbf{Canónico}}{\equiv} \\
GF\, D \wedge F \oplus A \sqsubseteq \oplus B & \overset{\textbf{EBin2}-\textbf{(a)}}{\equiv} \\
GF\, D \wedge F \oplus \oplus A \vee G\neg \oplus B & \overset{\textbf{Canónico}}{\equiv} \\
GF\, D \wedge (F \oplus^2 A \vee G \oplus \neg B) &
\end{array}
$$

Reuniendo las dos subfórmulas, obtenemos la forma normal negativa temporal de la fórmula original:

$$GF\,(\oplus C \sqsubset B) \wedge GF\, D \wedge (F \oplus^2 A \vee G \oplus \neg B)$$

10. Conclusiones y trabajos futuros

En este trabajo hemos partido de la lógica temporal de intervalos LNint-e$^+$ y hemos presentado el medio para reducir sus expresiones de intervalos a expresiones de puntos. Sobre LNint-e$^+$ se ha definido una nueva forma normal cuyo diseño ha sido guiado por la simplificación de las expresiones iniciales y que resulta independiente del método de deducción elegido.

Hemos diseñado un algoritmo que obtiene la forma normal negativa equivalente a una fórmula de entrada. La estructura básica del algoritmo consiste en la aplicación primero en profundidad de las leyes de la lógica clásica (doble negación, de Morgan, etc.) y de las transformaciones estudiadas en las secciones anteriores: **Canónico**, **TNeg**, **Dist**, **Ebin1**, **Ebin2**, **IConst** y **EConst**. La complejidad de este algoritmo es n^2 en el peor caso. Hemos implementado este algoritmo en Prolog.

A modo de ejemplo ilustrativo, se han llevado a cabo pruebas del algoritmo de normalización. Se realizó una generación aleatoria[16] de expresiones de LNint-e y sobre cada fórmula se ejecutó el procedimiento de normalización.

Los resultados en la reducción del tamaño de las fórmulas se desglosa en la tabla 1, en la que se detalla el porcentaje de reducción (en media) de símbolos proposicionales, conectivas booleanas, conectivas temporales monarias y binarias y del tamaño total de la fórmula.

	Reducción
Símbolos proposicionales	62 %
Conectivas booleanas	89,7 %
Conectivas monarias	49,9 %
Conectivas binarias	84,2 %
Tamaño total	59,9 %

Tabla 1. Resultados de las pruebas

[16]Se ha realizado una generación aleatoria de fórmulas debido a que no existen esquemas de fórmulas representativos para lógicas temporales totalmente expresivas (similares a los existentes para otras lógicas [29]).

Sobre estos resultados debemos destacar el alto porcentaje de eliminación de conectivas temporales binarias, que como ya se indicó en la introducción, era uno de los objetivos esenciales de este trabajo. Además, hemos de resaltar que casi un 47 % de las fórmulas se redujo a una constante ($\top$ o $\bot$), es decir, el algoritmo de normalización realizó completamente la demostración de la fórmula de entrada. Por último, también hemos de observar que aproximadamente un 16 % de las fórmulas aumentó su tamaño. La razón es que, en aquellas fórmulas cuya estructura es la de un prefijo de conectivas temporales monarias aplicado a una conjunción o disyunción, las leyes de distributividad "trasladan"este prefijo a cada una de las subfórmulas. En cualquier caso, esto no debe considerarse como un aspecto negativo, ya que se obtienen literales monarios con más información.

La forma normal obtenida es completamente coordinable con otras formas normales anteriormente introducidas y así puede ser incluida en los trabajos de M. Fisher [13, 19, 9] como paso previo a la normalización clausal para reducir el tamaño del problema original.

En [2] los autores demuestran que puede definirse una forma normal sobre la lógica temporal y luego usar el método de resolución clausal temporal presentado en [12] para obtener resultados más eficientes. Nuestro trabajo permite la depuración de las formas normales intermedias para ganar en simplicidad de estas expresiones sin aumentar complejidad en el método.

Referencias

[1] J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, jul 1984.

[2] A. Bolotov, M. Fisher, and C. Dixon. On the relationship between ω-automata and temporal logic normal forms. *Journal of Logic and Computation*, 12(4):561–581, 2002.

[3] A. Burrieza and I. P. de Guzmán. A new algebraic semantic approach and some adequate connectives for computation with temporal logic over discrete time. *Journal of Applied non Classical Logic*, 2, 1992.

[4] P. Cordero. *Ideales y filtros de implicantes/implicados en lógicas temporales con tiempo lineal y discreto.* PhD thesis, Universidad de Málaga, Spain, 1999.

[5] P. Cordero, M. Enciso, and I. P. de Guzmán. Bases for closed sets of implicants and implicates in temporal logic. *Acta Informatica*, 38:599–619, 2002.

[6] P. Cordero, M.Enciso, and I. Pérez de Guzmán. A temporal negative normal form which preserves implicants and implicates. *Journal of Applied Non-Classical Logics*, 10(3-4):243–272, 2000.

[7] I. Pérez de Guzmán and C. Rossi. LNint: a temporal logic that combines points and intervals and the absolute and relative approaches. *Journal of the IGPL*, 3(5), 1995.

[8] J. de Kleer, A. K. Mackworth, and R. Reiter. Characterizing diagnoses and systems. *Artificial Intelligence*, 56(2–3):197–222, 1992.

[9] C. Dixon, M. Fisher, and A. Bolotov. Clausal resolution in a logic of rational agency. *Artificial Intelligence*, 139(1):47–89, 2002.

[10] M. Enciso, J. M. Frías, and C. Rossi. Formalización de interfaces de usuario usando lógica temporal. In *INTERACCIÓN 2001, 2nd Congreso Internacional de Interacción Persona-Ordenador*, 2001.

[11] M. Enciso, I. P. de Guzmán, and C. Rossi. Using temporal logic to represent dynamic behaviour of UML statecharts. In *ECOOP 2002 Workshop on Integration and Transformation of UML Models*, 2002.

[12] M. Fisher. A resolution method for temporal logic. In *Proceedings of the 12^{th} International Joint Conference on Artificial Intelligence*, Sydney. Australia, 1991. Morgan Kaufmann.

[13] M. Fisher. A normal form for temporal logics and its applications in theorem proving and execution. *Journal of Logic and Computation*, 7(4):429–456, 1997.

[14] D. M. Gabbay. The declarative past and imperative future. In *Proccedings of the Colloquium on Temporal Logic and Specifications*, volume 398 of *Lecture Notes in Computer Science*, page 409—448. Springer Verlag, 1989.

[15] M. L. Ginsberg. A circumscriptive theorem prover. *Artificial Intelligence*, 39(2):209–230, 1989.

[16] E. Hajnicz. *Time Structures. Formal Description and Algorithmic Representation*, volume 1047 of *Lecture Notes in Artificial Intelligence*. Springer Verlag, 1996.

[17] J. Halpern and Y. Shoham. A propositional modal logic of time intervals. *Journal of the ACM*, 38(4):935–962, October 1991.

[18] B. A. Haugh. Non-standard semantics for the method of temporal arguments. In *Proceedings de la International Joint Conference on Artificial Intelligence IJCAI'87*, pages 449–455, 1987.

[19] U. Hustadt, C. Dixon, R.A. Schmidt, and M. Fisher. Normal forms and proofs in combined modal and temporal logics. In *Proceedings of the Third International Workshop on Frontiers of Combining Systems FroCoS'2000*, volume 1794 of *Lecture Notes in Artificial Intelligence*, pages 73–87. Springer Verlag, 2000.

[20] T. Ibaraki, A. Kogan, and K. Makino. Functional dependencies in horn theories. *Artificial Intelligence*, 108:1–30, 1999.

[21] H. Kamp. *Tense logic and theory of linear orders*. PhD thesis, University of california, Los Angeles, 1968.

[22] A. Kean. The approximation of implicates and explanations. *International Journal of Approximate Reasoning*, 9:97–128, 1993.

[23] F. Kröger. *Temporal logic of programs*. Springer-Verlag New York, Inc., 1987.

[24] P. Marquis. Extending abduction from propositional to first-order logic. In *Fundamentals of artificial intelligence research*, pages 141–155. Springer, 1991.

[25] D. V. McDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.

[26] D.A. Plaisted and S.A. Greenbaum. A structure-preserving clause form translation. *Journal Symbolic Computation*, 2(3):293–304, 1986.

[27] C. Rossi. *Lógica Temporal de Intervalos. Formalización de Diagramas de Estados*. PhD thesis, Universidad de Málaga, Spain, 2001.

[28] R. Socher. Optimizing the clausal normal form transformation. *Journal of automated reasoning*, 7(3):325–336, 1991.

[29] G. Sutcliffe and C. Suttner. The TPTP problem library for automated theorem proving, 2002. Disponible en `www.math.miami.edu/~tptp`.