UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE CIENCIAS MATEMÁTICAS
Departamento de Informática y Automática

TESIS DOCTORAL

Una lógica trivalorada para funciones recursivas parciales

MEMORIA PARA OPTAR AL GRADO DE DOCTOR
PRESENTADA POR

Antonio Gavilanes Franco

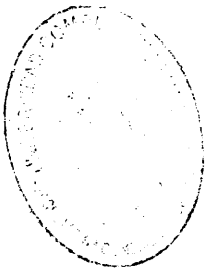
Madrid, 2015

UNIVERSIDAD COMPLUTENSE DE MADRID

Facultad de Ciencias Matemáticas

Departamento de Informática y Automática

UNA LOGICA TRIVALORADA PARA FUNCIONES RECURSIVAS PARCIALES



Antonio Gavilanes Franco

Madrid, 1990

Colección Tesis Doctorales. N.º 196/90

Edita e imprime la Editorial de la Universidad
Complutense de Madrid. Servicio de Reprografía
Escuela de Estomatología. Ciudad Universitaria
Madrid, 1990
Ricoh 3700
Depósito Legal: M-25287-1990

NC: X-53-164817-5

UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE CIENCIAS MATEMATICAS
DEPARTAMENTO DE INFORMATICA Y AUTOMATICA

UNA LOGICA TRIVALORADA PARA FUNCIONES RECURSIVAS PARCIALES

MEMORIA PRESENTADA POR
ANTONIO GAVILANES FRANCO

Para la Obtención del
GRADO DE DOCTOR EN CIENCIAS MATEMATICAS
Por la Universidad Complutense de Madrid

Bajo la Dirección de
MARIO RODRIGUEZ ARTALEJO

Madrid, Diciembre de 1989

¿Se acuerda de la manzana que Adán comió en el jardín del Edén, como se cuenta en la Biblia? -preguntó-. ¿Sabe lo que había en esa manzana?. Lógica. La lógica y demás cosas intelectuales. Eso es lo único que tenía dentro. Así que (esto es lo que quiero señalar) lo que tiene que hacer es vomitar todo eso si quiere ver las cosas como realmente son. Quiero decir que, si lo vomita, no va a tener más problemas con trozos de madera y cosas así. Ya no verá las cosas acabando todo el tiempo. Y sabrá qué es en realidad su brazo, si le interesa saberlo. ¿Comprende lo que quiero decir?. ¿Cree que lo ha entendido?.

J. D. Salinger
Teddy

Quiero expresar mi más profundo agradecimiento al director de este trabajo, Mario Rodríguez Artalejo, por el apoyo constante y la inmensa ayuda prestada durante estos años. Su inagotable buena voluntad y capacidad hicieron posible resolver, de forma sencilla, todos los problemas que se fueron presentando.

También quiero agradecer a Teresa Hortalá, la decidida ayuda y confianza que siempre me dispensó, así como su agudeza y memoria, de las que se benefició este trabajo; a Ana Gil, compañera de estudios y tareas desde hace mucho tiempo, la colaboración que me ha prestado desde entonces, desinteresadamente; a Paqui Lucio, a quien debo, amén de funciones complejas y comentarios provechosos, la buena exposición que se haya podido dar al tema; a Pepe Ruiz, quien nunca me dejó agradecerle su gran ayuda cuando había mucho más que tres valores; a todos los miembros de este Departamento, los que lo son y los que lo fueron, porque de todos aprendí algo que impulsó, de forma continuada, la labor durante este tiempo.

A mi familia, que ha padecido estos meses la fatigosa tarea de escribir este trabajo.

A todos, mis más sinceras gracias.

INDICE

INTRODUCCION

Una década antes de que Howard H. Aitken revolucionara el mundo de los computadores presentando su *Automatic Sequence Controlled Calculator* (ASCC), en agosto de 1944, en la Universidad de Harvard, haciendo realidad el sueño de Babbage de construir una máquina de calcular de uso general, un grupo de matemáticos había formalizado la noción de lo que se entendía por "computable" y había demostrado que, por encima de ella, había problemas que ni la misma ASCC lograría resolver nunca.

La forma de hacer esto es conocida. "Basta", como hizo Turing, idear una sencilla máquina, que realice unas sencillas operaciones, para superar en potencia, y quizá en rapidez, los engranajes de las 750000 piezas y las conexiones de los más de 800000 metros de cable de la ASCC y, aunque se tardaran más de los diez segundos que empleaba ella en realizar una división (algo complicada), se tenía al alcance de la mano la posibilidad de cualquier cómputo que se pudiera pensar.

Desde entonces hasta hoy las máquinas reales han aumentado extraordinariamente su potencia, mientras que, en la vertiente teórica, se ha investigado, entre otros temas, el comportamiento de mecanismos imaginarios más proximos a ellas, no tan potentes, en ciertos casos, como para poder alcanzar a definir todo lo computable. Se han estudiado así, sistemas formales que permiten razonar sobre las propiedades que tienen los programas que los gobiernan: corrección, terminación, equivalencia y síntesis de programas, potencia expresiva de operadores, etc.

Surgieron así, a finales de la década de los sesenta, diversos sistemas, más o menos relacionados con la lógica matemática, que hacían hincapié en alguno de los temas anteriores. Son de este tipo los primeros trabajos de Engeler y Thiele, el método de las aserciones invariantes de Floyd y Naur, las lógicas de las que fueron precursores, como la algorítmica de Salwicki, la modal de Pratt, la dinámica de Harel, Meyer y Pratt, la temporal de Pnueli, la de procesos, ... Y cada una de estas últimas definidas, a su vez, sobre un conjunto diverso de programas, con un conjunto de posibilidades de cómputo más o menos próximo a la realidad de un lenguaje imperativo de alto nivel. Para más detalles pueden consultarse los trabajos monográficos sobre el tema de [KT] y [Har 84].

La práctica común en estas lógicas era entender un programa como una transformación de unos datos en otros; con otras palabras, entender el significado de un programa como una función de los valores de entrada que

recibía, sobre los valores de salida que producía, lo que suele denominarse función de entrada/salida.

Sin embargo, mientras que, en la realidad, un programa podía no llegar a producir ningún resultado a causa, la mayor parte de las veces, de las limitaciones físicas del ordenador o de las intelectuales del que lo programaba (valores fuera de rango, errores, excepciones, funciones recursivas divergentes, iteraciones sin fin, desbordamiento, etc.), generalmente los sistemas lógicos que los estudiaban consideraban la función de entrada/salida, asociada a un programa, como total.

El tratamiento de las funciones parciales dentro de la lógica matemática no era nuevo cuando aparecieron los primeros trabajos que se ocupaban del tema desde un punto de vista computacional. Sin embargo, para los lógicos, en principio, la motivación tuvo un carácter más matemático. Si una función parcial divergía en un conjunto conocido de valores, era sencillo convertirla en total o, más aún, ocuparse solamente del grafo que definía.

Desde un punto de vista computacional, en cambio, el problema es distinto. La posibilidad de averiguar cuándo un programa arbitrario diverge es uno de esos problemas que se sabe no tiene solución, desde hace más de medio siglo. Por tanto no basta solamente con convertir, a nivel semántico, las funciones parciales en totales o considerar su grafo.

Como solución, una de las posturas que se puede tomar, y así lo haremos, es hacer algo más que lo anterior. Para ello, en primer lugar convertiremos las funciones parciales en totales haciendo que valgan un elemento distinguido nuevo en aquellos argumentos en que divergen. La forma de hacer esto es, siguiendo una conocida idea de Scott [Sco 70], considerar las funciones, definidas sobre cpo's, reservando su elemento mínimo, para elemento distinguido. A su vez, el sistema formal dispondrá de un mecanismo sintáctico para razonar únicamente en los casos en los que la función converge.

Ahora bien, cabe preguntarse qué sentido tiene una fórmula que contiene expresiones parcialmente definidas -es decir, cuyo valor semántico es el elemento mínimo de un cierto cpo. La respuesta, por nuestra parte, ha sido considerar un nuevo valor booleano *indefinido*, además de los clásicos *cierto* y *falso*. De esta manera, una fórmula que no contiene suficicente información como para poder determinar su certeza o falsedad, queda indefinida; por tanto el nuevo valor booleano actúa como elemento mínimo de información en la semántica y, en consecuencia, pueden tratarse, de la misma forma, las

funciones parciales y los predicados parciales, ya que en ambos casos reservamos un elemento para la divergencia. Además, extendemos la aplicación del operador de definición desde las expresiones a las fórmulas de manera que el sistema formal disponga de mecanismos sintácticos para razonar solo en los casos en que el predicado converge.

Este operador de definición y este sentido semántico, que empleamos para funciones y predicados parciales, constituyeron la base del presente trabajo en el que trataremos de estudiar mecanismos teóricos de cómputo, más o menos potentes y más o menos cercanos a la realidad de un lenguaje de programación, y en el que nos ocuparemos, fundamentalmente, del *carácter parcial* de las funciones que los rigen.

La otra característica esencial de este trabajo es el *marco funcional puro* en el que se desarrolla. Las lógicas que mencionábamos más arriba se han centrado más en lenguajes de programación imperativos. En su lugar emplearemos una combinación del λ-cálculo con la lógica de predicados, con la que se pueden expresar propiedades de funciones obtenidas por recursión.

Este trabajo está estructurado en cuatro capítulos, divididos en secciones, y unos apéndices finales que los complementan. Hagamos un breve repaso de lo tratado en cada uno de ellos.

El primer capítulo está dedicado a aspectos relacionados con la introducción de un nuevo valor booleano y todos ellos caen dentro del campo de lo que se dio en llamar *lógica multivalorada*. Se trata de determinar cómo se pueden extender ciertas nociones desde la lógica clásica (en adelante, el adjetivo *clásico* hará referencia al caso en que se disponga solamente de dos valores booleanos: cierto y falso) hasta la lógica trivalorada. Dichas nociones incluyen fundamentalmente las que se refieren a conectivas y cuantificadores y la de relación de consecuencia lógica, concluyendo el capítulo con la elección de un sentido para cada una de ellas. En concreto, la consecuencia lógica es aquella que permite el paso de hipótesis ciertas a conclusiones no falsas. Ya que no es transitiva, los cálculos que obtengamos no podrán serlo tampoco, con lo que optaremos por los de secuencias.

El segundo capítulo es un estudio del primer sistema formal que presentamos y está basado en el de [GL 89]. Se trata de una lógica de primer orden para funciones parciales *LFP* con construcciones condicionales para los términos. Los cómputos estudiados son pues limitados. Sin embargo, el capítulo servirá para investigar, más adelante, posibilidades de cómputo

mayores.

Mediante el método de los tableaux mostraremos cómo obtener condiciones de forma que los cálculos de secuencias que las satisfagan sean completos. A su vez, la completitud del método de los tableaux se basará en una construcción "al estilo de Hintikka", adaptada a las peculiaridades del lenguaje.

El tercer capítulo se ocupa de extender el sistema formal anterior. Se trata en él una lógica de primer orden para funciones recursivas parciales *LFRP*. El apelativo "primer orden" hace referencia al hecho de que las funciones que manejamos tienen un tipo, construido sobre un conjunto de géneros básicos, de forma que los argumentos que toma la función son todos de género básico. Las funciones de las que nos ocupamos se obtienen mediante dos operadores: uno, el usual de abstracción del λ-cálculo, y otro, el de recursión.

El desarrollo del capítulo es similar al anterior. La semántica considera las funciones como continuas sobre dominios de estructuras arbitrarias de primer orden. Empleamos el método de los tableaux para obtener condiciones de completitud de forma que los cálculos de secuencias que las satisfagan sean completos. En este caso, sin embargo, los cálculos son infinitarios y, aunque, como para *LFP*, sigue siendo cierto el teorema de Skolem-Löwenheim, la lógica pierde su carácter de primer orden y no satisface el teorema de compacidad.

Por último, el capítulo cuarto está dedicado a exponer una aplicación de la lógica *LFRP* al estudio de la semántica de un lenguaje de programación imperativo *LI*. Este lenguaje contiene las construcciones usuales de uno tipo ALGOL, incluyendo las de declaración local de variables y procedimientos, eventualmente recursivos con paso de parámetros por valor y por referencia. Utilizamos *LFRP* como especificación semántica de los programas de *LI*, asociando a cada uno de estos, una expresión del primero; de esta forma el lenguaje de programación imperativo queda dotado con una semántica de tipo denotacional, como la que se da para *LFRP*, conceptualmente más clara y simple. Además mostramos cómo es posible traducir las fórmulas de una lógica de programas para *LI* a fórmulas de nuestra lógica, poniendo de manifiesto que ésta dispone de tanta potencia expresiva como las lógicas de programas al uso, para lenguajes imperativos.

CAPITULO 1
LOGICAS 3-VALORADAS

El tratamiento de las funciones parciales y, por ende, la noción de valor indefinido de una función para un argumento dado, puede incluirse, y de hecho históricamente así ha ocurrido, en el marco más general de verdad *imprecisa* (cuando no es posible obtener la certeza o falsedad de una fórmula si una parte suya no tiene valor veritativo completo, cierto o falso), *incierta* (cuando el valor veritativo de una fórmula "puede llegar a hacerse" cierto o falso, con idéntica probabilidad en uno u otro caso) o *indeterminada* (cuando el valor veritativo de una fórmula queda determinado por la respuesta de un proceso de cómputo cualquiera).

Estos conceptos que, por diversos motivos, han estado presentes en el trabajo de muchos investigadores desde principios de siglo, han confluido en lo que ha venido en llamarse *lógica multivalorada* ("many-valued logic").

Básicamente se trata de formalizar el razonamiento bajo los tipos de verdad, arriba mencionados, añadiendo uno o más valores veritativos a los clásicos *cierto* y *falso*, pero manteniendo la idea de la lógica clásica de que el valor veritativo de una fórmula depende del de sus componentes (extensionalidad).

En la primera parte de este capítulo nos ocupamos de las lógicas con tres valores veritativos -ya que, como quedará patente en capítulos posteriores, bastará añadir un nuevo valor booleano para poder tratar la divergencia de las funciones- y presentamos, en primer lugar, de qué forma se puede ampliar el sentido de las conectivas proposicionales cuando interviene un nuevo valor booleano, haciendo un repaso histórico de los intentos que ha habido al respecto y optando finalmente por las conectivas definidas por Kleene. En la siguiente sección veremos cómo es posible ampliar estas conectivas a fin de obtener un conjunto funcionalmente completo; esta noción se obtiene, de manera obvia, a partir de la clásica, es decir, se trata de un conjunto de conectivas en función del cual puede obtenerse por composición cualquier otra conectiva extensional. Terminaremos con un estudio de los cuantificadores análogo al hecho con las conectivas, optando por aquellos que extiendan el caso finito definido por éstas; es decir, una cuantificación existencial (resp. universal) será entendida como una disyunción (resp. conjunción) infinita. Esto nos llevará a elegir también la semántica de Kleene para los cuantificadores.

En la segunda parte del capítulo estudiaremos cómo afecta la introducción de un nuevo valor booleano a ciertos conceptos semánticos. Así, veremos primeramente que existen varias maneras de definir la noción de modelo de una fórmula: modelos fuertes (aquellos que la hacen cierta), débiles (los que no la hacen falsa) e indefinidos (los que la hacen indefinida); a cada una de estas nociones se asocia a su vez otra de satisfactibilidad.

Hecho ésto estaremos en disposición de definir la relación de consecuencia lógica en una forma análoga a como se hace en el caso clásico. Ya que existen tres maneras de satisfacer las hipótesis y otras tres de hacerlo con la conclusión, se presentarán hasta nueve formas de definir la consecuencia lógica. En la sección segunda se definen éstas y se hace un estudio de las relaciones que existen entre ellas.

A pesar de que este proceso parece llevar a una definición general de la noción de consecuencia, veremos en la siguiente sección que algunos trabajos sobre lógicas 3-valoradas emplean nociones que no se contemplan entre las definidas. Como nuestro objetivo en los capítulos siguientes es obtener sistemas formales correctos y completos, basados en tres valores booleanos, la importancia de la consecuencia lógica elegida será muy grande. Es por ésto que en la sección cuarta nos proponemos un estudio de este concepto desde un punto de vista absolutamente general, que permita englobar a las formas de consecuencia lógica más importantes que se han tratado.

Dicho estudio nos lleva a considerar todas las formas de modelo que admiten las hipótesis de un razonamiento -fuerte, débil e indefinido- y dar para cada una de ellas -si hace el caso- el tipo de modelo que se admite en la conclusión. La diferencia con el enfoque anterior es que se puede definir con más finura el paso de unas a otra; así, podremos admitir que la conclusión no sea falsa, si no lo son las hipótesis, pero exigiendo que sea cierta, si sí lo son.

Dedicamos las dos secciones siguientes a aplicar este concepto general al caso 3-valorado y a hacer un estudio de las características principales que presentan las relaciones de consecuencia lógica resultantes. Dicho estudio permitirá concluir que las consecuencias que satisfacen propiedades importantes de la lógica clásica y que, por tanto, se prestan mejor para ser empleadas en capítulos posteriores, son las que se obtienen por nuestra primera definición, en la cual relacionábamos el tipo de modelo de las hipótesis y la conclusión, de manera fija.

Concluido el capítulo, dispondremos de los principales conceptos semánticos necesarios para el tratamiento de cualquier sistema formal sobre tres valores veritativos, en particular una lógica para funciones parciales. Dichos conceptos son, básicamente, un significado para las conectivas y cuantificadores y una forma de consecuencia lógica.

A. SIMBOLOS LOGICOS EN LOGICAS 3-VALORADAS.

1.- CONECTIVAS 3-VALORADAS.

Por lógica multivalorada, en lo que sigue m-valorada, entendemos cualquier sistema què formalice el razonamiento, en el sentido clásico, usando m valores booleanos (o veritativos): cierto y falso, más m-2 valores, para reflejar situaciones de indefinitud, indeterminación, imprecisión, incertidumbre, etc.

Como veremos, a nosotros nos bastará restringirnos al caso m=3. Una visión histórica del desarrollo de las lógicas 3-valoradas mostrará la base para construir sistemas formales en los que poder tratar las funciones parciales.

Aunque aparecen referencias en Boole, Peirce y Vasiliev, puede decirse que la era moderna de las lógicas m-valoradas empieza con los primeros trabajos de Łukasiewicz [Łuk 20] y Post [Pos 21].

Al primero le guían motivos filosóficos. Su objetivo es liberarse de la "coerción lógica" que supone considerar a todas las sentencias plenas de verdad o falsedad. Para ello añade a los dos valores booleanos tradicionales, cierto y falso, un nuevo valor que expresa "lo posible" y que, por tanto, se moverá entre lo verdadero y lo falso. Siguiendo la notación de Łukasiewicz, tendremos la lógica 3-valorada sobre {0, 1/2, 1} con el orden numérico que existe entre ellos.

Basándose en {0, 1/2, 1}, Łukasiewicz define dos conectivas extensionales, la negación y la implicación, a través de las siguientes tablas veritativas:

	¬
0	1
1/2	1/2
1	0

→	0	1/2	1
0	1	1	1
1/2	1/2	1	1
1	0	1/2	1

Dos observaciones importantes hay que hacer sobre estas conectivas. En primer lugar, se comportan como las clásicas cuando sus argumentos son valores veritativos clásicos. En segundo lugar, una implicación se hace cierta cuando antecedente y consecuente no están definidos (esto es, toman ambos el valor 1/2). Como señala Blikle [Bli 88], cualquier lógica con

funciones parciales que esté basada en esta interpretación no será implementable. En efecto, la no terminación es una propiedad no computable y, por tanto no se podrá decidir, en general, cuando los miembros de una implicación toman valor indefinido.

Caso diferente fue el de Post quien, siguiendo consideraciones matemáticas, definió una lógica con m valores veritativos $M=\{0,1,\ldots,m-1\}$ y dos conectivas:

$$\neg x = (x+1) \bmod m$$

$$xvy = \text{minimo } \{x,y\}$$

para $x, y \in M$. El orden en M es el aritmético, pero la interpretación de sus elementos es la contraria de Łukasiewicz; ahora 0 se refiere a lo cierto y m-1, a lo falso.

Investigaciones sobre la teoría de las funciones recursivas llevaron a Kleene a definir una lógica basada en tres valores veritativos {t, i, f} [Kle 52]. En este caso i puede imaginarse como la "respuesta" que daría una máquina si, bajo ciertas entradas, no parara por cualquier motivo.

En este enfoque, las tablas veritativas, para la negación y disyunción, son:

	¬
t	f
i	i
f	t

v	t	i	f
t	t	t	t
i	t	i	i
f	t	i	f

Como se comprueba fácilmente, la disyunción es la de Post para el caso m=3 y la negación es la de Łukasiewicz.

Si se piensa en la implicación como su abreviatura clásica es inmediato ver que la implicación de Kleene es igual que la de Łukasiewicz salvo en el crítico valor $i \rightarrow i$ que ahora es indefinido. Incluso bajo el supuesto de que interpretemos i como "lo posible" parece claro que si el antecedente y el consecuente pueden llegar a ser ciertos o falsos, a la implicación le ocurre lo mismo y es, por tanto, lógico concluir que su valor esté indefinido.

Sin embargo, la interpretación usual de las conectivas de Kleene se enmarca en la teoría de la computabilidad. Así, la respuesta de una máquina que calcula el valor de una disyunción será "cierto" tan pronto obtenga "cierto" como valor de alguno de los miembros de la disyunción, mientras que su respuesta no se hará "definida" en tanto alguno de ellos no se haga "cierto".

No obstante, no es ésta la única manera de interpretar computacionalmente unas conectivas. Kleene imagina una máquina "paralela" que evalúa a la vez ambos miembros de una disyunción. Podemos, sin embargo, pensar en una máquina "secuencial" que evalúe primero el izquierdo y después el derecho. De esta manera, si el primer miembro no se hiciera "definido", la disyunción tampoco, independientemente de lo que le ocurriera al segundo.

Es éste el caso de las conectivas de McCarthy [McC 63] definidas por:

	¬
t	f
i	i
f	t

v	t	i	f
t	t	t	t
i	i	i	i
f	t	i	f

Decimos entonces que la disyunción es estricta por la izquierda. Con ello queremos expresar únicamente el hecho, arriba mencionado, de que ivx=i, para todo x∈{t, i, f}. Si representamos dicha disyunción por .v es inmediato definir las disyunciones estrictas por la derecha v. y por ambos lados .v. mediante las tablas veritativas:

v.	t	i	f
t	t	i	t
i	t	i	i
f	t	i	f

.v.	t	i	f
t	t	i	t
i	i	i	i
f	t	i	f

Las conectivas estrictas por ambos lados reflejan la idea de que cualquier fórmula que contenga una componente no definida queda indefinida. Sobre ellas basa Bochvar [Boc 39] su lógica 3-valorada.

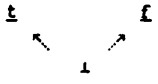
Como vemos los tres tipos de verdad, mencionados en la introducción de este capítulo, quedan capturados por las interpretaciones estudiadas: la verdad incompleta por Bochvar, la incierta por Łukasiewicz y la indeterminada por Kleene.

En adelante nosotros optaremos por las conectivas de Kleene, a las que nos referiremos mediante K (={¬, v}), entendiendo que otras conectivas extensionales como ∧, → y ↔, se interpretan como abreviaturas en función de los elementos de K, a la manera tradicional.

Así mismo supondremos definido el conjunto de valores veritativos por:

1.-Definición.

B=(B, ≤) es el orden formado por el conjunto de tres valores booleanos B={$\underline{t}$, $\underline{f}$, ⊥}, que corresponden a los clásicos *cierto* y *falso*, más un valor *indefinido*, respectivamente, parcialmente ordenados por ≤ de acuerdo con el diagrama:



■

Aunque volveremos sobre el tema en el capítulo siguiente, destacar un hecho importante: el orden dado a los valores booleanos convierte a **B** en un orden parcial completo (abreviado, cpo) plano (cf. [Ber 85]). Tales órdenes sirven para modelizar funciones parciales (cf. [Sco 70]), utilizando su elemento mínimo para expresar la indefinición o divergencia.

De las conectivas K de Kleene definidas sobre B destacamos las siguientes propiedades:

2.-Lema.

(a) K restringido a {$\underline{t}$, $\underline{f}$} coincide con la negación y la disyunción clásicas.

(b) La disyunción v entendida como función binaria de B^2 en B (resp. la negación ¬ de B en B), es una función monótona; esto es:

$$b \leq b' \Rightarrow avb \leq avb',\ bva \leq b'va \text{ (resp. } \neg b \leq \neg b')$$

para todo $a, b, b' \in B$.

(c) En las mismas condiciones que en (b), la función v no es estricta en ninguno de sus argumentos.

Demostración. (a) es inmediato.

(b) Si b≤b' entonces si b=b', es evidente. Si no, b=⊥ (pues el orden es plano) con lo cual:

(i) Si a=$\underline{f}$ o a=⊥ entonces avb=⊥≤avb'.

(ii) Si a=$\underline{t}$ entonces avb=avb'=$\underline{t}$.

Para bva≤b'va basta observar que v es conmutativa. Por último, la negación se comprueba de forma inmediata.

(c) Recordar los comentarios sobre las conectivas .v, v. y .v.. ■

2.- COMPLETITUD FUNCIONAL.

El objetivo de esta sección es encontrar un conjunto de conectivas C tal que K⊆C y C sea funcionalmente completo, concepto éste que tiene el mismo sentido que en lógica clásica, es decir:

1.-Definición.

Un conjunto A de conectivas se dice *funcionalmente completo* para un conjunto de valores booleanos arbitrario B si para cualquier función n-aria $F:B^n\to B$, $n\geq 0$, existe una función n-aria $G:B^n\to B$, obtenida a partir de los elementos de A mediante composición, tal que $F(\bar{a}) = G(\bar{a})$, para toda n-tupla $\bar{a}\in B^n$. ■

Nosotros buscamos la completitud funcional con respecto al conjunto B definido en 1.1. Para ello fijamos nuestra atención primeramente en las conectivas monádicas. Para abreviar la escritura, adoptemos el convenio de asociar $\underline{t}$ con 0, $\perp$ con 1, $\underline{f}$ con 2 y $\mathbb{T}$ con el conjunto {0, 1, 2}. Es inmediato que las conectivas $\neg$ y $\vee$ de K se corresponden entonces con las funciones: $h_0(x)=2-x$ y $h_1(x,y)=$mínimo{x, y}, respectivamente.

Para demostrar que K no es funcionalmente completo, consideremos las funciones $f_i:\mathbb{T}\to\mathbb{T}$, i=0,1,2, definidas por $f_i(x)=(x+i) \bmod 3$.

2.-Lema.

No existe ninguna función n-aria $\varphi:\mathbb{T}^n\to\mathbb{T}$, $n\geq 0$, obtenida a partir de h_0 y h_1 mediante composición, de forma que $\varphi(1,..,1)=0$ o $\varphi(1,..,1)=2$.

Demostración. Por inducción sobre el número k de apariciones de h_0 y h_1 en la definición de φ. Para k=1 es evidente. Supuesto para k entonces para k+1 caben las siguientes posibilidades:

(i) $\varphi(\bar{x})=h_0(\psi(\bar{x}))$. Entonces, si $\varphi(1,..,1)=0$ (resp. 2), tendremos que $\psi(1,..,1)=2$ (resp. 0) y ψ tendría a lo sumo k apariciones de h_0 y h_1, contra la hipótesis de inducción.

(ii) $\varphi(\bar{x})=h_1(\psi_1(\bar{x}),\psi_2(\bar{x}))$. Entonces:

$\varphi(1,..,1)=h_1(\psi_1(1,..,1),\psi_2(1,..,1))=0$ (resp. 2) ⇒

⇒ $\psi_1(1,..,1)=0$ o $\psi_2(1,..,1)=0$ (resp. $\psi_i(1,..,1)=2$, i=1,2)

contradiciendo de nuevo la hipótesis de inducción. ■

3.-Corolario.

No es posible expresar a partir de h_0 y h_1, mediante composición, las funciones f_1 y f_2.

Demostración. Basta observar que $f_1(1)=2$ y $f_2(1)=0$. ■

Obsérvese que K permite expresar, en cambio, fórmulas nunca falsas, como $p \vee \neg p$, o nunca ciertas, tales como $p \wedge \neg p$, donde p representa una variable proposicional. Estos ejemplos guardan relación con las leyes del *tercio excluso* y la *contradicción*, respectivamente.

Consideremos ahora la conectiva monádica Δ definida por la tabla veritativa:

	Δ
$\underline{t}$	$\underline{t}$
$\underline{f}$	$\underline{t}$
$\perp$	$\underline{f}$

Se trata del que llamaremos *operador de definición*, ya que se hace cierto cuando su argumento no está indefinido y falso, cuando lo está. En términos numéricos podemos escribirla como:

$$h_2(x)=\begin{cases} 0 & \text{si } x\neq 1 \\ 2 & \text{si } x=1 \end{cases}$$

Según acabamos de ver, Δ no es expresable en términos de las conectivas de K. Aún así, el conjunto $\{\neg, \vee, \Delta\}$ sigue sin ser funcionalmente completo.

4.-Lema.

No existe ninguna función n-aria $\varphi: \mathbb{T}^n \rightarrow \mathbb{T}$, $n\geq 0$, obtenida a partir de h_i, $i=0,1,2$, por composición, de forma que $\varphi(0,..,0)=1$.

Demostración. Por 1.2.(a), h_0 y h_1 restringidas a $\{0,2\}$ tienen rango $\{0,2\}$ y, según acabamos de ver, el rango de h_2 es también $\{0,2\}$. Por tanto, cualquier función χ, obtenida a partir de h_i, $i=0,1,2$, por composición, verifica que, restringida a $\{0,2\}$, nunca toma valor 1. ■

Por tanto, una fórmula siempre indefinida no es expresable en términos de las conectivas $\{\neg, \vee, \Delta\}$. Sin embargo, sí se puede expresar una fórmula nunca indefinida; en efecto, basta recordar que el rango de h_2 es $\{0,2\}$ y tomar la fórmula Δp. Como consecuencia de ello es posible obtener fórmulas siempre ciertas, como $\Delta\Delta p$, o siempre falsas, como $\neg\Delta\Delta p$ (algo que, según hemos visto

en 2, no era posible hacer con $\{\neg, \vee\}$ únicamente).

Consideremos pues una conectiva que permita expresar una fórmula siempre indefinida. Para ello introducimos la siguiente notación.

Por T (resp. F, Ω) representamos la conectiva 0-ádica que siempre da como valor veritativo $\underline{t}$ (resp. $\underline{f}$, $\perp$). Según sabemos, podemos asociar a Ω la función 0-ádica h_3: $\longrightarrow \mathbb{T}$ definida por: $h_3(\ ) = 1$.

Dado un conjunto de funciones A definidas sobre $\mathbb{T}$ con rango en $\mathbb{T}$ representamos por L_A el conjunto:

$\bigcup_{n\in\mathbb{N}} \{\varphi: \mathbb{T}^n \longrightarrow \mathbb{T} \ / \ \varphi$ es una función n-aria obtenida a partir de funciones de A por composición$\}$

5.-Teorema.

(a) $L_{(h_0,h_1)} \subset L_{(h_0,h_1,h_2)}$

(b) $L_{(h_0,h_1)} \subset L_{(h_0,h_1,h_3)}$

(c) $L_{(h_0,h_1,h_2)}$ y $L_{(h_0,h_1,h_3)}$ son incomparables.

Demostración. (a) Por 2: $h_2 \notin L_{(h_0,h_1)}$.

(b) Por 4: $h_3 \notin L_{(h_0,h_1,h_2)}$.

(c) Como en (b): $L_{(h_0,h_1,h_3)}$ no está contenido en $L_{(h_0,h_1,h_2)}$. Extendiendo el lema 2 al conjunto $\{h_0, h_1, h_3\}$ es inmediato demostrar que $h_2 \notin L_{(h_0,h_1,h_3)}$. ■

Terminamos esta sección demostrando la completitud funcional del conjunto de conectivas $\{\neg, \vee, \Delta, \Omega\}$. Para abreviar la escritura consideraremos las funciones h y g asociadas a la conjunción e implicación, respectivamente. Ya que $p \wedge q$ es la abreviatura de $\neg(\neg p \vee \neg q)$, se tiene $h(x,y) = 2\text{-mínimo } \{2-x, 2-y\}$. De forma similar, como $p \longrightarrow q$ representa $\neg p \vee q$ entonces $g(x,y) = \text{mínimo } \{2-x, y\}$.

6.-Teorema.

Dada una función n-aria, $n \geq 0$, $\varphi: \mathbb{T}^n \longrightarrow \mathbb{T}$, $\varphi \in L_{(h_0,h_1,h_2,h_3)}$.

Demostración. Seguiremos una demostración similar a la de la lógica proposicional clásica (cf. [Dal 80]) y haremos inducción sobre n.

El caso n=0 es trivial ya que solo hay tres conectivas extensionales 0-ádicas: la propia Ω, F, que puede definirse mediante $\Delta\Omega$, y T, que se obtiene con $\neg\Delta\Omega$.

Supuesto para $n\geq 0$, sea $\varphi:\mathbf{T}^{n+1}\to\mathbf{T}$ y condideremos las funciones n-arias $\varphi_k:\mathbf{T}^n\to\mathbf{T}$, k=0,1,2, definidas por:

$\varphi_k(x_1,\ldots,x_n) = \varphi(x_1,\ldots,x_n,k)$.

Por hipótesis de inducción existen $\psi_k \in L_{(h_0,h_1,h_2,h_3)}$, k=0,1,2, tales que: $\psi_k(x_1,\ldots,x_n) = \psi_k(\bar{x}) = \varphi_k(\bar{x})$, k=0,1,2. Consideremos la función $H:\mathbf{T}^{n+1}\to\mathbf{T}$ definida por:

$$H(x_1,\ldots,x_n,x_{n+1}) = H(\bar{x}, x_{n+1}) = h[\, g(h(x_{n+1}, h_2(x_{n+1})), \psi_0(\bar{x})),$$
$$h(\, g(h(h_0(x_{n+1}), h_2(x_{n+1})), \psi_2(\bar{x})),$$
$$g(h_0(h_2(x_{n+1})), \psi_1(\bar{x})))]$$

La expresión booleana anterior se corresponde con la siguiente fórmula lógica:

$$(p_{n+1}\wedge\Delta p_{n+1}\to F_0(\bar{p})) \wedge (\neg p_{n+1}\wedge\Delta p_{n+1}\to F_2(\bar{p})) \wedge (\neg\Delta p_{n+1}\to F_1(\bar{p}))$$

donde F_i es la conectiva n-aria que tiene por función asociada a ψ_i, i=0,1,2.

Se tiene que $H=\varphi$ puesto que:

(i) $x_{n+1}=0 \Rightarrow H(\bar{x},0) = h[g(h(0,0),\psi_0(\bar{x})),$
$h(g(h(2,0),\psi_2(\bar{x})),g(2,\psi_1(\bar{x})))] =$
$= h(g(0,\psi_0(\bar{x})),h(g(2,\psi_2(\bar{x})),0)) =$
$= h(\psi_0(\bar{x}),h(0,0)) = h(\psi_0(\bar{x}),0) =$
$= \psi_0(\bar{x}) = \varphi_0(\bar{x}) = \varphi(\bar{x},0)$

(ii) $x_{n+1}=1 \Rightarrow H(\bar{x},1) = h[g(h(1,2),\psi_0(\bar{x})),$
$h(g(h(1,2),\psi_2(\bar{x})),g(0,\psi_1(\bar{x})))] =$
$= h(g(2,\psi_0(\bar{x})),h(g(2,\psi_2(\bar{x})),\psi_1(\bar{x}))) =$
$= h(0,h(0,\psi_1(\bar{x}))) = h(0,\psi_1(\bar{x})) =$
$= \psi_1(\bar{x}) = \varphi_1(\bar{x}) = \varphi(\bar{x},1)$

(iii) $x_{n+1}=2 \Rightarrow H(\bar{x},2) = h[g(h(2,0),\psi_0(\bar{x})),$
$h(g(h(0,0),\psi_2(\bar{x})),g(2,\psi_1(\bar{x})))] =$
$= h(g(2,\psi_0(\bar{x})),h(g(0,\psi_2(\bar{x})),0)) =$
$= h(0,h(\psi_2(\bar{x}),0)) = h(0,\psi_2(\bar{x})) =$
$= \psi_2(\bar{x}) = \varphi_2(\bar{x}) = \varphi(\bar{x},2)$

■

Igual que al término de la sección precedente supusimos fijo el conjunto de conectivas K de Kleene, de ahora en adelante supondremos dado el conjunto de conectivas $C\mathbb{K} = K \cup \{\Delta, \Omega\}$, funcionalmente completo.

3.- CUANTIFICADORES 3-VALORADOS.

Como se ha visto en la sección primera de este capítulo, existen varias maneras de definir el significado de las conectivas cuando se refieren a valores booleanos no clásicos. Otro tanto puede decirse de los cuantificadores si se interpretan como conjunciones y disyunciones infinitas.

Si siguiéramos la línea expuesta en 1, un repaso histórico del tema nos mostraría dos formas principales de definir la semántica de los cuantificadores, basadas en su "carácter estricto".

Por tal "carácter" entendemos lo siguiente. Según se vio en 1.2.(c), la disyunción de Kleene no es estricta en ninguno de sus argumentos. Si imaginamos la cuantificación existencial como una disyunción infinita podemos definir su semántica como [Kle 52]:

$$\mathfrak{I}(\exists x\varphi) = \begin{cases} \underline{t} & \text{si existe } a \in A \text{ tal que } \mathfrak{I}[a/x](\varphi)=\underline{t} \\ \underline{f} & \text{si todo } a \in A \text{ verifica que } \mathfrak{I}[a/x](\varphi)=\underline{f} \\ \perp & \text{en otro caso} \end{cases}$$

donde $\mathfrak{I}$ es una interpretación, formada por una estructura $\mathfrak{A}$, de dominio A, y una valoración de variables v, e $\mathfrak{I}[a/x]$ es la interpretación que coincide con $\mathfrak{I}$ salvo en x, que se valora como a.

Observar que una fórmula $\exists x\varphi$ queda indefinida si algún valor de x hace indefinida a φ, y el resto no la hace cierta.

Por tanto la semántica del cuantificador existencial vuelve a ser "no estricta" en el sentido de que $\exists x\varphi$ se puede hacer cierta, a pesar de que para algún valor de x la subfórmula φ no esté definida.

Expresando la cuantificación universal $\forall x\varphi$ como su abreviatura clásica $\neg\exists x\neg\varphi$, resulta:

$$\mathfrak{I}(\forall x\varphi) = \begin{cases} \underline{t} & \text{si todo } a \in A \text{ verifica que } \mathfrak{I}[a/x](\varphi)=\underline{t} \\ \underline{f} & \text{si existe } a \in A \text{ tal que } \mathfrak{I}[a/x](\varphi)=\underline{f} \\ \perp & \text{en otro caso} \end{cases}$$

Como antes, es inmediato ver que $\forall x\varphi$ queda indefinida si algún valor de x hace indefinida a φ, y el resto no la hace falsa.

Si tomamos las conectivas de McCarthy (cf. sección 1) y generalizamos su definición a un número no finito de miembros resulta:

$\mathfrak{I}(\exists x\varphi) = \underline{t}$ ⇔ existe $a\in A$ tal que $\mathfrak{I}[a/x](\varphi)=\underline{t}$ y todo $a\in A$ verifica que $\mathfrak{I}[a/x](\varphi)\neq\perp$

$\mathfrak{I}(\exists x\varphi) = \underline{f}$ ⇔ todo $a\in A$ verifica que $\mathfrak{I}[a/x](\varphi)=\underline{f}$

quedando indefinida en otro caso; es decir, cuando algún valor de x hace indefinida a φ. Es por ello que esta definición se corresponde con una interpretación "estricta" de la disyunción.

Como hemos venido haciendo en secciones anteriores, elegimos la semántica de Kleene para los cuantificadores. Por tanto, en lo que queda de capítulo, supondremos fijo:

(1) un conjunto de conectivas básico K, formado por la disyunción y la negación de Kleene

(2) dos conectivas auxiliares que añadidas a K proporcionan un conjunto funcionalmente completo CK: el operador de definición Δ y la conectiva siempre indefinida Ω

(3) la semántica "no estricta" de Kleene para los cuantificadores, que extiende de forma natural la dada para la disyunción y la conjunción.

Los apartados que siguen forman, en su mayor parte, un estudio que no depende de las elecciones que acabamos de mencionar. En aquellas partes en que no sea así (generalmente, resultados que dependen de la existencia de ciertas fórmulas con un valor semántico determinado), la metodología allí seguida no variará si se hacen otras elecciones de conectivas y únicamente serán necesarios mínimos cambios.

B. CONCEPTOS SEMANTICOS EN LOGICAS 3-VALORADAS.

1.- MODELOS Y SATISFACTIBILIDAD EN LOGICAS 3-VALORADAS.

Ciertos conceptos semánticos que en lógica clásica están determinados de manera unívoca, deben ser precisados cuando se añade un nuevo valor booleano. Tal es el caso de la noción de modelo. Recordemos que en aquélla, dados un conjunto de fórmulas Φ y una interpretación $\mathfrak{I}$, se dice que $\mathfrak{I}$ es modelo de Φ si todas sus fórmulas son ciertas en $\mathfrak{I}$; es decir, si $\mathfrak{I}(\varphi)=\underline{t}$, para toda $\varphi\in\Phi$.

Ahora bien, en lógicas 3-valoradas, una fórmula que no es falsa no siempre será cierta con lo cual la noción de modelo debería hacer referencia al hecho de que todas las fórmulas de Φ se hagan ciertas, o parte de ellas queden indefinidas. Caben entonces tres tipos de modelo.

1.-<u>Definición</u>. *Modelo fuerte, débil e indefinido.*

Dados un conjunto de fórmulas Φ y una interpretación $\mathfrak{I}$ decimos que:

(1) $\mathfrak{I}$ es *modelo fuerte* de Φ, lo que escribimos mediante $\mathfrak{I}\models_s\Phi$, si y solo si $\mathfrak{I}(\varphi)=\underline{t}$, para toda fórmula $\varphi\in\Phi$.

(2) $\mathfrak{I}$ es *modelo débil* de Φ, lo que escribimos mediante $\mathfrak{I}\models_w\Phi$, si y solo si $\mathfrak{I}(\varphi)\neq\underline{f}$, para toda fórmula $\varphi\in\Phi$.

(3) $\mathfrak{I}$ es *modelo indefinido* de Φ, escrito $\mathfrak{I}\models_u\Phi$, si y solo si $\mathfrak{I}(\varphi)=\perp$, para toda fórmula $\varphi\in\Phi$. ■

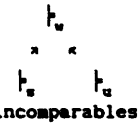
Los conceptos de fórmula e interpretación se precisarán en el siguiente capítulo.

Como se ve hay tres nociones de modelo correspondiendo a las tres posibilidades que existen cuando ninguna fórmula del conjunto es falsa; a saber, que todas sean ciertas, que lo sea alguna o que no lo sea ninguna.

La definición anterior se generaliza al caso m-valorado de forma inmediata. Basta decir, dado un conjunto cualquiera Bool de "valores booleanos" y dado $D\subseteq Bool$, un subconjunto suyo no vacío tal que $\underline{f}\in D$, que una interpretación $\mathfrak{I}$ es un *D-modelo* de Φ, escrito $\mathfrak{I}\models_D\Phi$, si y solo si $D\supseteq\mathfrak{I}(\Phi)$, donde $\mathfrak{I}(\Phi) = \{\mathfrak{I}(\varphi)/\varphi\in\Phi\}$. Obsérvese que esto comprende los casos antes definidos tomando $s=D_s=\{\underline{t}\}$ para $\models_s$, $w=D_w=\{\underline{t}, \perp\}$ para $\models_w$ y $u=D_u=\{\perp\}$ para $\models_u$. Obviamente, si $\mathfrak{I}$ es tal que $\mathfrak{I}(\varphi)\neq\underline{f}$, para toda $\varphi\in\Phi$, entonces $\mathfrak{I}$ es un

$\mathfrak{I}(\Phi)$-modelo de Φ.

Los conceptos de modelo que acabamos de definir, considerados como relaciones de modelo, esto es, relaciones binarias entre interpretaciones y conjuntos de fórmulas, cumplen el siguiente diagrama:



(incomparables)

donde las flechas indican inclusión estricta, debido a las inclusiones estrictas $\{\underline{t}\}$, $\{\iota\} \subset \{\underline{t}, \iota\}$ y a la existencia de las fórmulas T, Ω.

Como en lógica clásica, cada tipo de modelo origina una forma de satisfactibilidad. Recordemos que en aquélla, un conjunto de fórmulas Φ se dice satisfactible si existe una interpretación $\mathfrak{I}$ que hace ciertas todas las fórmulas de Φ; o, equivalentemente, que no las hace falsas. Siguiendo un análisis similar al hecho más arriba, podemos hablar de un conjunto de fórmulas Φ *fuertemente satisfactible*, escrito $Sat_s(\Phi)$, *débilmente satisfactible*, escrito $Sat_w(\Phi)$, o *indefinidamente satisfactible*, escrito $Sat_u(\Phi)$.

2.- RELACIONES DE CONSECUENCIA EN LOGICAS 3-VALORADAS.

La noción de consecuencia lógica en lógica clásica se formula en función del concepto de modelo. Así, dado un conjunto de fórmulas $\Phi \cup \{\varphi\}$ se dice que φ es consecuencia lógica de Φ, y se escribe $\Phi \models \varphi$, si para toda interpretación $\mathfrak{I}$, si $\mathfrak{I}$ es modelo de Φ entonces también lo es de φ. Siguiendo una notación habitual en lógica de primer orden (cf. [EFT 84]) y representando por $Mod(\Phi)$ el conjunto de interpretaciones que son modelo de Φ, lo anterior puede ser escrito como $Mod(\Phi) \subseteq Mod(\{\varphi\})$.

Ya que en lógicas 3-valoradas existen tres tipos de modelo, podrían darse, en principio, nueve formas de consecuencia lógica.

1.-Definición.

Dados un conjunto de fórmulas $\Phi \cup \{\varphi\}$ y $a, b \in \{s, w, u\}$, decimos que φ es *ab-consecuencia lógica* de Φ, y escribimos $\Phi \models_{ab} \varphi$, si y solo si para toda interpretación $\mathfrak{I}$ se tiene: $\mathfrak{I} \models_a \Phi \Rightarrow \mathfrak{I} \models_b \varphi$. ■

Siguiendo un estudio similar al que suele hacerse en lógica clásica nos ocuparemos ahora de dos temas: i) la relación que existe entre los conceptos de modelo, satisfactibilidad y consecuencia que hemos definido, y ii) la conexión de las ab-consecuencias entre sí.

Respecto de i), como es sabido, en lógica clásica se tiene la equivalencia:

1) $\Phi \vdash \varphi$

2) $Mod(\Phi) \subseteq Mod(\varphi)$

3) $\Phi \cup \{\neg\varphi\}$ no es satisfactible

En nuestro caso, la equivalencia de 1) y 2) para $\vdash_{ab}$ es análoga definiendo $Mod_a(\Phi)$, $a \in \{s,w,u\}$, como el conjunto de interpretaciones que son a-modelo de Φ.

Para la equivalencia de 1) y 3) distinguimos el valor de $b \in \{s,w,u\}$. Por ejemplo, para w tenemos:

2.-<u>Lema</u>.

Dado un conjunto de fórmulas $\Phi \cup \{\varphi\}$:

(a) $\Phi \vdash_{sw} \varphi \Leftrightarrow \Phi \cup \{\neg\varphi\}$ no es s-satisfactible.

(b) $\Phi \vdash_{ww} \varphi \Leftrightarrow \Phi \cup \{\neg\varphi \wedge \Delta\varphi\}$ no es w-satisfactible.

(c) $\Phi \vdash_{uw} \varphi \Leftrightarrow \Phi \cup \{\neg\varphi \wedge \Delta\varphi \wedge \Omega\}$ no es u-satisfactible.

<u>Demostración</u>. Basta aplicar las definiciones de consecuencia dadas, teniendo en cuenta las siguientes observaciones:

para (a): $\mathfrak{I}(\varphi)=\underline{f} \Leftrightarrow \mathfrak{I}(\neg\varphi)=\underline{t}$

para (b): $\mathfrak{I}(\varphi)=\underline{f} \Leftrightarrow \mathfrak{I}(\neg\varphi \wedge \Delta\varphi)\neq\underline{f}$

para (c): $\mathfrak{I}(\varphi)=\underline{f} \Leftrightarrow \mathfrak{I}(\neg\varphi \wedge \Delta\varphi \wedge \Omega)=\perp$. ∎

Obsérvese en el lema anterior que las fórmulas T, Ω (y F) están bien formadas debido a que fueron definidas como conectivas 0-ádicas (cf. A.2).

De forma similar, y aprovechando la completitud funcional del conjunto de conectivas CK, es fácil encontrar fórmulas φ_i, i=1,2,3, en función de φ de manera que:

$\mathfrak{I}(\varphi)\neq\perp \Leftrightarrow \mathfrak{I}(\varphi_1)=\underline{t}$

$\mathfrak{I}(\varphi)\neq\perp \Leftrightarrow \mathfrak{I}(\varphi_2)\neq\underline{f}$

$\mathfrak{I}(\varphi)\neq\perp \Leftrightarrow \mathfrak{I}(\varphi_3)=\perp$

Tomar, por ejemplo, $\Delta\varphi$, $\Delta\varphi$ y $\Delta\varphi \wedge \Omega$, respectivamente. Dichas fórmulas permiten expresar las consecuencias lógicas $\vdash_{au}$, $a \in \{s,w,u\}$, en función de la

a-satisfactibilidad.

Finalmente, para relacionar $\vdash_{as}$ con la a-satisfactibilidad basta encontrar fórmulas φ_i, i=1,2,3, tales que:

$\mathfrak{I}(\varphi)\neq\underline{t} \Leftrightarrow \mathfrak{I}(\varphi_1)=\underline{t}$

$\mathfrak{I}(\varphi)\neq\underline{t} \Leftrightarrow \mathfrak{I}(\varphi_2)\neq\underline{f}$

$\mathfrak{I}(\varphi)\neq\underline{t} \Leftrightarrow \mathfrak{I}(\varphi_3)=\perp$

Para lo primero bastaría con encontrar una conectiva unaria $H(\varphi)$ de manera que: $\varphi=\perp \Rightarrow H(\varphi)=\underline{t}$, $\varphi=\underline{f} \Rightarrow H(\varphi)=\underline{t}$ y $\varphi=\underline{t} \Rightarrow H(\varphi)\neq\underline{t}$; se puede elegir, entonces $\Delta\varphi\rightarrow\neg\varphi$. Análogamente los otros dos casos.

Respecto de ii), esto es, la conexión de las ab-consecuencias entre sí, cabe decir que cada una de ellas puede ponerse en función de cualquiera de las otras, como demuestran los siguientes dos lemas.

3.-<u>Lema</u>.

Para todo conjunto de fórmulas $\Phi \cup \{\varphi\}$ se tiene:

(a) $\Phi \vdash_{ub} \varphi \Leftrightarrow \{\neg\Delta\psi \;/\; \psi\in\Phi\} \vdash_{sb} \varphi$.

(b) $\Phi \vdash_{sb} \varphi \Leftrightarrow \{\psi\wedge\Delta\psi \;/\; \psi\in\Phi\} \vdash_{wb} \varphi$.

(c) $\Phi \vdash_{wb} \varphi \Leftrightarrow \{\Delta\psi\rightarrow\psi \;/\; \psi\in\Phi\} \vdash_{sb} \varphi$.

(d) $\Phi \vdash_{sb} \varphi \Leftrightarrow \{\psi\wedge\Delta\psi\wedge\Omega \;/\; \psi\in\Phi\} \vdash_{ub} \varphi$.

<u>Demostración</u>. Comprobación inmediata teniendo en cuenta que:

$\mathfrak{I}(\psi)=\perp \Leftrightarrow \mathfrak{I}(\neg\Delta\psi)=\underline{t}$

$\mathfrak{I}(\psi)=\underline{t} \Leftrightarrow \mathfrak{I}(\psi\wedge\Delta\psi)\neq\underline{f}$

$\mathfrak{I}(\psi)\neq\underline{f} \Leftrightarrow \mathfrak{I}(\Delta\psi\rightarrow\psi)\,(=\mathfrak{I}(\neg\Delta\psi\vee\psi))=\underline{t}$

$\mathfrak{I}(\psi)=\underline{t} \Leftrightarrow \mathfrak{I}(\psi\wedge\Delta\psi)\neq\underline{f} \Leftrightarrow \mathfrak{I}(\psi\wedge\Delta\psi\wedge\Omega)=\perp$. ■

4.-<u>Lema</u>.

Para todo conjunto de fórmulas $\Phi \cup \{\varphi\}$ se tiene:

(a) $\Phi \vdash_{au} \varphi \Leftrightarrow \Phi \vdash_{as} \neg\Delta\varphi$.

(b) $\Phi \vdash_{as} \varphi \Leftrightarrow \Phi \vdash_{au} \Delta\varphi\wedge\varphi$.

(c) $\Phi \vdash_{au} \varphi \Leftrightarrow \Phi \vdash_{as} \Delta\varphi\rightarrow\varphi$.

(d) $\Phi \vdash_{as} \varphi \Leftrightarrow \Phi \vdash_{au} \varphi\wedge\Delta\varphi\wedge\Omega$.

<u>Demostración</u>. Análoga al lema anterior. ■

Las ab-consecuencias lógicas que acabamos de presentar son, sin embargo, demasiado generales en el sentido de que algunas de ellas son incompatibles con la clásica. Así por ejemplo, $\vdash_{ub}$, $b \in \{s,w,u\}$, admitiría que $T \vdash_{ub} F$. Igualmente, para $\vdash_{au}$, $a \in \{s,w,u\}$, se da el caso de que no es cierto que $\vdash_{au} T$. Optaremos pues por restringirnos a aquellas consecuencias $\vdash_{ab}$ que extienden la clásica en el doble sentido que acabamos de precisar: no se tiene $T \vdash_{ab} F$, pero sí $\vdash_{ab} T$. Esto supone considerar únicamente las siguientes formas de ab-consecuencia: $\vdash_{ws}$, $\vdash_{ww}$, $\vdash_{ss}$ y $\vdash_{sw}$.

Teniendo en cuenta las relaciones que existen entre los modelos fuerte y débil, es inmediato demostrar:

5.-<u>Lema</u>.

Dado un conjunto de fórmulas $\Phi \cup \{\varphi\}$:

$$\begin{array}{ccc} \Phi \vdash_{ww} \varphi & \Longrightarrow & \Phi \vdash_{sw} \varphi \\ \Uparrow & & \Uparrow \\ \Phi \vdash_{ws} \varphi & \Longrightarrow & \Phi \vdash_{ss} \varphi. \end{array}$$

Además, $\vdash_{ww}$ y $\vdash_{ss}$ son incomparables.

<u>Demostración</u>. Para demostrar que $\vdash_{ww}$ y $\vdash_{ss}$ son incomparables observar que $T \vdash_{ww} \Omega$, pero no para $\vdash_{ss}$. Igualmente $\Omega \vdash_{ss} F$, pero no para $\vdash_{ww}$. ∎

De este lema deducimos que $\vdash_{ws}$ es la relación más fuerte, en el sentido de que implica a las otras tres, y $\vdash_{sw}$ es la más débil, en el sentido de que es implicada por las otras tres.

Antes de terminar conviene hacer una observación importante sobre los conceptos definidos en esta última sección. Se ha estudiado la relación de consecuencia lógica para pares de la forma (Φ, φ). La generalización a pares (Φ, Ψ), donde Φ y Ψ son conjuntos cualesquiera de fórmulas, puede hacerse entendiendo el antecedente Φ como una conjunción y el consecuente Ψ como una disyunción. De esta manera, decimos que $\Phi \vdash_{ab} \Psi$ si para toda interpretación $\mathfrak{I}$, si $\mathfrak{I}$ es un a-modelo de *todas* las fórmulas de Φ entonces $\mathfrak{I}$ es un b-modelo de *alguna* fórmula de Ψ. Los lemas 2, 3 y 4 se extienden a conjuntos de fórmulas de manera natural. En particular, se verifica:

$$\Phi \vdash_{sw} \Psi \Leftrightarrow \Phi \cup \{\neg\psi \;/\; \psi \in \Psi\} \text{ no es s-satisfactible}$$

3.- OTRAS RELACIONES DE CONSECUENCIA LOGICA.

Antes de describir de qué forma es posible tratar de determinar la relación de consecuencia lógica óptima y en qué sentido ésta lo es, daremos un breve repaso a algunos trabajos previos sobre lógicas 3-valoradas con el fin de expresar las consecuencias, en ellos usadas, en función de los conceptos aquí definidos.

La mayor parte de las relaciones $\vdash_{ws}$, $\vdash_{ww}$, $\vdash_{ss}$ y $\vdash_{sw}$, que obtuvimos en la sección anterior han sido la base de algunos trabajos sobre lógicas 3-valoradas. Así, una consecuencia que impusiera conclusiones ciertas a partir de hipótesis ciertas, $\vdash_{ss}$, fue usada por Kleene [Kle 52] en sus estudios sobre el tema; más tarde sería empleada para el estudio formal de las funciones parciales por Barringer et al. [BCJ 84], aunque introduciendo el operador de definición Δ. Por motivos similares fueron las investigaciones de Hoogewijs [Hoo 87], quien empleó la relación débil $\vdash_{sw}$.

La llamada "relación paraconsistente" que Avron define en sus trabajos [Avr 87], [Avr 88] permite sacar conclusiones no falsas a partir de hipótesis no falsas, es decir, $\vdash_{ww}$. Se trata así mismo de la que Owe [Owe 84] emplea para estudiar la que, él llama, "weak logic", un formalismo parecido al de primer orden para el estudio de las funciones parciales.

No conocemos ningún trabajo en el cual se haya tratado con la relación más fuerte de las cuatro, $\vdash_{ws}$; se trata de una consecuencia verdaderamente extraña que no siempre permite concluir una fórmula φ a partir de sí misma (observar que, por ejemplo, no es cierto $\Omega\vdash_{ws}\Omega$).

El repertorio de consecuencias usadas en trabajos sobre lógicas 3-valoradas no termina, sin embargo, con las mencionadas hasta ahora. Existen otras más o menos extrañas, distintas de las anteriores, que muestran que el concepto de ab-consecuencia definido más arriba no es lo suficientemente general.

Tal es el caso de la tratada por Blamey [Bla 86] en su estudio sobre lógicas parciales. Allí se define:

$\Phi\vdash_{Bla}\varphi$ ⇔ para toda interpretación J:

1) $J(\psi)=\underline{t}$, para toda $\psi\in\Phi$ ⇒ $J(\varphi)=\underline{t}$

2) $J(\varphi)=\underline{f}$ ⇒ $J(\psi)=\underline{f}$, para alguna $\psi\in\Phi$

A partir de esta definición es inmediato comprobar que:

$$\Phi\vdash_{Bla}\varphi \Leftrightarrow \Phi\vdash_{ss}\varphi \text{ y } \Phi\vdash_{ww}\varphi$$

con lo que también se tiene:

$$\Phi \vdash_{ws} \varphi \Rightarrow \Phi \vdash_{Bla} \varphi \Rightarrow \Phi \vdash_{sw} \varphi$$

por el lema 2.5.

Otra consecuencia que no se cubre hasta ahora es la utilizada por Sobocinski [Sob 52] definida por:

$\Phi \vdash_{Sob} \varphi \Leftrightarrow$ para toda interpretación $\mathfrak{I}$:

$\mathfrak{I}(\psi) \neq \underline{f}$, para toda $\psi \in \Phi \Rightarrow [\mathfrak{I}(\varphi) = \underline{t}$ o $\mathfrak{I}(\eta) = \perp$, para toda $\eta \in \Phi \cup \{\varphi\}]$

Es decir, o bien alguna hipótesis es falsa, o bien la conclusión es cierta o bien hipótesis y conclusión son todas indefinidas. Esta relación presenta una patología curiosa; en efecto, con ella no es posible debilitar el antecedente pues se tiene $\Omega \vdash_{Sob} \Omega$, pero no Ω, $T \vdash_{Sob} \Omega$.

No parece que la consecuencia de Sobocinski sea expresable como una combinación sencilla de consecuencias de la forma $\vdash_{ab}$, $a, b \in \{s, w, u\}$. Lo mismo ocurre para la consecuencia de Łukasiewicz, cuya definición, siguiendo [Avr 87], es:

$\Phi \vdash_{\text{Łuk}} \varphi \Leftrightarrow$ para toda interpretación $\mathfrak{I}$:

$\mathfrak{I}(\psi) \neq \underline{f}$, para toda $\psi \in \Phi \Rightarrow [\mathfrak{I}(\varphi) = \underline{t}$ o al menos dos fórmulas de $\Phi \cup \{\varphi\}$ están indefinidas en $\mathfrak{I}]$

En esta definición, la intención es que $\Phi \cup \{\varphi\}$ sea tratado como multiconjunto, de manera que las posibles apariciones repetidas de fórmulas cuenten según su número. Se consigue así que valga $\Omega \vdash_{\text{Łuk}} \Omega$, y en general, $\varphi \vdash_{\text{Łuk}} \varphi$ para cualquier φ. En cambio, no valen $T \vdash_{\text{Łuk}} \Omega$ ni $\Omega \vdash_{\text{Łuk}} F$, mientras que $\{\Omega, \Omega\} \vdash_{\text{Łuk}} F$ sí sería cierto.

Este comportamiento nos parece muy patológico, por lo cual no intentaremos que los métodos a desarrollar en las siguientes secciones abarquen consecuencias como ésta.

4.- GERMENES DE CONSECUENCIA LOGICA.

Vista la existencia de relaciones de consecuencia lógica que no son del tipo $\vdash_{ab}$, nos ocuparemos ahora de investigar maneras sistemáticas de definirlas, considerando las consecuencias lógicas como relaciones entre dos conjuntos de fórmulas Φ y Ψ, entendidos como *antecedente* y *consecuente*.

Un procedimiento general para definir relaciones de consecuencia en lógicas m-valoradas consiste en emplear alguna variante del llamado método de la matriz (cf. [Urq 86]). En él, dados un conjunto B de m valores booleanos y un subconjunto suyo D no vacío tal que no contiene el valor falso, cuyos

elementos son llamados destacados, una consecuencia lógica $\vdash$ es una relación entre conjuntos de fórmulas de tal forma que:

$\Phi \vdash \Psi$ ⇔ para toda interpretación $\mathfrak{I}$:

$\mathfrak{I}(\varphi) \in D$, para toda $\varphi \in \Phi$ ⇒ $\mathfrak{I}(\psi) \in D$, para alguna $\psi \in \Psi$

De esto se deduce que todas las consecuencias lógicas definidas a través del método de la matriz satisfacen las siguientes dos propiedades:

(1) Reflexividad: $\Phi \vdash \Psi$, si $\Phi \cap \Psi \neq \varnothing$.

(2) Transitividad (o corte): si $\Phi \vdash \Psi \cup \{\varphi\}$ y $\Phi' \cup \{\varphi\} \vdash \Psi'$ entonces $\Phi \cup \Phi' \vdash \Psi \cup \Psi'$.

La demostración de que satisfacen (1) es inmediata. Para (2), supongamos que $\mathfrak{I}$ es una interpretación tal que $\mathfrak{I}(\eta) \in D$, para toda $\eta \in \Phi \cup \Phi'$; si $\mathfrak{I}(\psi) \notin D$ para ninguna $\psi \in \Psi \cup \Psi'$, entonces $\mathfrak{I}(\varphi) \in D$, pues $\Phi \vdash \Psi \cup \{\varphi\}$. Pero $\Phi' \cup \{\varphi\} \vdash \Psi'$, luego existe $\psi \in \Psi'$ tal que $\mathfrak{I}(\psi) \notin D$, contra lo que acabamos de suponer.

De forma alternativa, podríamos haber definido las relaciones de consecuencia lógica como aquéllas que verifican (1) y (2). De hecho, así se hace en [Avr 87] [Avr 88].

No obstante, el método de la matriz no permite distinguir entre una relación que satisfaga $\Omega \not\vdash \Omega$ y otra que no lo haga.

Al tratar de superar este inconveniente, una primera aproximación nos lleva a considerar dos conjuntos de valores booleanos de elementos destacados, D_1 y D_2, para poder referirnos al valor respectivo que toman antecedente y consecuente. Por tanto, como antes, supondremos que D_1 y D_2 son conjuntos no vacíos que no contienen el valor booleano falso. Podemos definir entonces:

$\Phi \vdash \Psi$ ⇔ para toda interpretación $\mathfrak{I}$:

$\mathfrak{I}(\varphi) \in D_1$, para toda $\varphi \in \Phi$ ⇒ $\mathfrak{I}(\psi) \in D_2$, para alguna $\psi \in \Psi$

Si una relación de consecuencia $\vdash$ es definida así entonces:

(3) $\vdash$ es reflexiva ⇔ $D_1 \subseteq D_2$.

Para la demostración de ésta y propiedades similares, en adelante supondremos que en una lógica con un conjunto Bool de m valores booleanos cualesquiera se dispone de un conjunto de conectivas funcionalmente completo tal que dado $D \subseteq Bool$, existe un conjunto de fórmulas Φ_D construido sobre ellas tal que $\mathfrak{I}(\Phi_D) = D$ (cf. comentarios a la definición 1.1 para la notación $\mathfrak{I}(\Phi_D)$), para toda interpretación $\mathfrak{I}$. En el caso 3-valorado que venimos estudiando, ello ocurre así, según se demostró en A.2, considerando T, F y Ω. Si D es un conjunto formado por un solo valor booleano $a \in Bool$ entonces escribiremos φ_a para expresar que $\Phi_{(a)} = \{\varphi_a\}$.

Supongamos entonces que $\vdash$ es reflexiva y sea $a \in D_1$; entonces $\varphi_a \vdash \varphi_a$ luego $a \in D_2$. El recíproco es obvio.

(4) $\vdash$ es transitiva $\Leftrightarrow D_2 \subseteq D_1$.

Supongamos que $\vdash$ es transitiva y que D_2 no está incluido en D_1. Existirá pues un elemento $a \in D_2 \setminus D_1$; llegamos a contradicción eligiendo $b \in D_1$, que existe por ser D_1 no vacío, y observando que $\varphi_b \vdash \varphi_a$ y $\varphi_a \vdash F$, pero no $\varphi_b \vdash F$, pues el valor booleano falso no pertenece a D_2. El recíproco se sigue con un razonamiento similar al dado cuando se demostró (2).

De esta forma se pueden conseguir relaciones de consecuencia que sean o no sean reflexivas y/o transitivas. Sin embargo, si buscamos una lógica 3-valorada que verifique (a) $\Omega \vdash \Omega$, (b) no se tenga $\Omega \vdash F$ y además extienda la clásica en el sentido de que no permita que (c) $T \vdash F$, entonces $\underline{t} \in D_1$ (por (c)), $\iota \in D_1$ (por (b)) y $\iota \in D_2$ (pues $\iota \in D_1$ y (a)). Por tanto no es posible conseguir que $\vdash$ no satisfaga $\{\Omega, T\} \vdash \Omega$, con lo que $\vdash$ no será la consecuencia de Sobocinski. Parece necesario pues distinguir casos y definir explícitamente cómo transforma $\vdash$ cada tipo de modelo al pasar de las premisas a la conclusión.

Recordemos que en lógicas m-valoradas existía una noción de modelo por cada subconjunto de valores booleanos $D \subseteq Bool$ no vacío que no contuviera el valor falso (cf. comentarios a la definición 1.1). Un primer intento de definición nos lleva a considerar relaciones que asocien a cada subconjunto $D \subseteq Bool$ un tipo de modelo determinado por otro conjunto similar $D' \subseteq Bool$. Es decir, una relación de consecuencia podría ser entonces un subconjunto ρ de $\mathcal{P}(Bool) \times \mathcal{P}(Bool)$ de manera que:

$\Phi \vdash \Psi \Leftrightarrow$ para toda interpretación $\mathfrak{I}$:

$\mathfrak{I}$ es D-modelo de Φ, $(D, D') \in \rho \Rightarrow \mathfrak{I}$ es un D'-modelo de alguna fórmula de Ψ

Ahora bien, conviene exigir que si $(D, D') \in \rho$ entonces $D' \neq Bool$ pues en otro caso $\mathfrak{I}$ siempre sería un D'-modelo de alguna fórmula de Ψ. En las mismas condiciones es lógico pedir que $D' \neq \emptyset$ pues en otro caso $\mathfrak{I}$ nunca podría ser D'-modelo de ninguna fórmula de Ψ.

También sería deseable que el conjunto vacío se comportara semánticamente siempre como un conjunto de fórmulas fuertemente válidas, esto es, ciertas en toda interpretación (recordar que, por definición, $\emptyset$ es también indefinidamente satisfactible). De este modo, si $(D, D') \in \rho$ entonces podemos suponer que $D \neq \emptyset$, pues lo que ocurre cuando el antecedente es vacío es lo mismo que cuando está formado por fórmulas fuertemente válidas y, como

veremos, ρ siempre dirá algo, si éste es el caso.

Dicho esto se puede definir cuándo un subconjunto de $\wp(\text{Bool})\times\wp(\text{Bool})$ determina una relación de consecuencia lógica.

1.-Definición. *Gérmenes de consecuencia lógica.*

Un germen de consecuencia lógica es una relación $\rho\subseteq\wp(\text{Bool})\times\wp(\text{Bool})$ tal que:

para todo A, $C\in\wp(\text{Bool})$: $((A, C)\in\rho \Rightarrow \varnothing \neq A,\ \varnothing \neq C \neq \text{Bool})$. ■

En adelante escribiremos la pertenencia a gérmenes, $(A, C)\in\rho$, mediante notación infija, $A\rho C$.

Según los comentarios que acabamos de hacer sobre la forma en que esperamos se transformen los D-modelos, podemos definir la relación de consecuencia lógica asociada, a un germen de la forma siguiente:

2.-Definición. *Consecuencia lógica inducida por un germen.*

Dados un conjunto de fórmulas Φ y una interpretación $\mathfrak{I}$ cualesquiera definimos el valor booleano de Φ bajo $\mathfrak{I}$ en el antecedente, escrito $A_{\mathfrak{I}}(\Phi)$, y en el consecuente, escrito $C_{\mathfrak{I}}(\Phi)$, respectivamente mediante:

$$A_{\mathfrak{I}}(\Phi) = \begin{cases} \mathfrak{I}(\Phi) & \text{si } \Phi \neq \varnothing \\ \{\underline{t}\} & \text{en otro caso} \end{cases}$$

y, dualmente:

$$C_{\mathfrak{I}}(\Phi) = \begin{cases} \mathfrak{I}(\Phi) & \text{si } \Phi \neq \varnothing \\ \{\underline{f}\} & \text{en otro caso} \end{cases}$$

Dado un germen ρ, definimos la consecuencia lógica $\vdash_\rho$ inducida por ρ como una relación binaria entre conjuntos de fórmulas, de la manera siguiente:

$\Phi \vdash_\rho \Psi \Leftrightarrow$ Para toda interpretación $\mathfrak{I}$ y todo $C\subseteq B$:

$$(A_{\mathfrak{I}}(\Phi)\rho C \Rightarrow C\ni C_{\mathfrak{I}}(\Psi))$$

donde $C'\ni C \Leftrightarrow C' \cap C \neq \varnothing$. ■

Dados D, $D'\subseteq\text{Bool}$ tales que $D\subseteq D'$ se tiene, según sabemos, que todo D-modelo es un D'-modelo. Obsérvese, sin embargo, que un germen ρ tal que $D\rho C$ puede verificar $D'\rho C'$, para algún C' disjunto con C. Es decir, una consecuencia $\vdash_\rho$ que muestre cómo se transforma un D'-modelo, no explica implícitamente cómo se transforma un D-modelo, siendo $D\subseteq D'$. Es por ello que a la hora de definir gérmenes se ha de tener ésto en cuenta, como quedará

patente más adelante.

Las ab-consecuencias que definimos en la sección 2 son inducidas por gérmenes, como pasamos a demostrar. Recordemos que en la sección 1 introdujimos las siguientes notaciones: s={$\underline{t}$}, u={$\bot$} y w={$\underline{t}$, $\bot$}. Sea, por ejemplo, la ss-consecuencia. En ella se transforma todo modelo fuerte en un modelo fuerte. Por tanto, el germen ρ_{ss}={(s,s)} induce $\vdash_{ss}$. Análogamente puede demostrarse que:

(1) $\vdash_{sw} = \vdash_{\rho_{sw}}$ donde ρ_{sw}={(s,w)}

(2) $\vdash_{ws} = \vdash_{\rho_{ws}}$ donde ρ_{ws}={(s,s), (u,s), (w,s)}

(3) $\vdash_{ww} = \vdash_{\rho_{ww}}$ donde ρ_{ww}={(s,w), (u,w), (w,w)}

(4) $\vdash_{Bla} = \vdash_{\rho_{Bla}}$ donde ρ_{Bla}={(s,s), (u,w), (w,w)}

(5) $\vdash_{Sob} = \vdash_{\rho_{Sob}}$ donde ρ_{Sob}={(s,s), (u,w), (w,s)}.

Obsérvese que $\vdash_{Sob}$ exige transformar un modelo débil en uno fuerte, pues $(w,s)\in\rho_{Sob}$, mientras que permite transformar uno indefinido, que también lo es débil, en uno débil, pues $(u,w)\in\rho_{Sob}$. Sobre estas y otras propiedades hablaremos en la siguiente sección. Antes terminaremos haciendo notar que las relaciones de consecuencia obtenidas por el método de la matriz son también inducidas por gérmenes. En efecto, dado $\emptyset \subset D \subset$ Bool tal que $\underline{f}\notin D$ se tiene que la relación $\vdash_D$, definida por D de acuerdo con lo dicho al principio de esta sección, coincide con $\vdash_{\rho_D}$ donde ρ_D={(D',D) / $\emptyset\subset D'\subseteq D$}. De la misma manera, si distinguimos dos conjuntos de valores booleanos D_1, D_2 con $\emptyset \subset D_1, D_2 \subset$ Bool entonces la relación de consecuencia $\vdash_{D_1,D_2}$ que definen coincide con $\vdash_{\rho_{D1,D2}}$ donde ρ_{D_1,D_2}={(D', D_2) / $\emptyset \subset D'\subseteq D_1$}.

5.- PROPIEDADES DE LAS CONSECUENCIAS LOGICAS INDUCIDAS POR GERMENES.

Terminábamos la sección anterior haciendo notar que ciertos gérmenes de consecuencias lógicas, como ρ_{Sob}, no satisfacen una propiedad de "monotonía" según la cual si ρ transforma modelos débiles en fuertes entonces ha de transformar, en particular, los indefinidos también en fuertes. Parece lógico imponer ésta y otras propiedades que pasamos ahora a formular en términos de restricciones sobre los gérmenes. Antes de ello introduzcamos la siguiente notación. Dado un germen ρ, llamamos dominio y rango de ρ a los conjuntos definidos por:

$dom(\rho) = \{A \subseteq Bool \ / \text{ existe } C \subseteq Bool \text{ tal que } A \rho C\}$

$ran(\rho) = \{C \subseteq Bool \ / \text{ existe } A \subseteq Bool \text{ tal que } A \rho C\}$.

respectivamente.

1.-Definición. *Gérmenes y consecuencias lógicas monótonos.*

Un germen ρ se dice monótono si y solo si para todos A, A', $C' \subseteq Bool$ se verifica que:

$\varnothing \subset A \subset A' \rho C' \Rightarrow [A \in dom(\rho)$ y para todo C: $A \rho C \Rightarrow C \subseteq C']$

Una consecuencia lógica $\vdash$ se dice monótona si y solo si para cualesquiera conjuntos de fórmulas Φ_1, Φ_2, Ψ_1, Ψ_2 se verifica que:

$\Phi_1 \vdash \Psi_1$, $\Phi_1 \subseteq \Phi_2$, $\Psi_1 \subseteq \Psi_2 \Rightarrow \Phi_2 \vdash \Psi_2$. ■

Es evidente que todos los gérmenes definidos al final de la sección anterior son monótonos, salvo el de Sobocinski, ya que (w,s), $(u,w) \in \rho$.

Veamos la relación existente entre la monotonía de un germen y la monotonía de la consecuencia lógica que éste induce.

2.-Lema.

Si un germen ρ es monótono entonces la consecuencia lógica que induce, $\vdash_\rho$, es monótona.

Demostración. Supongamos que $\Phi_1 \vdash_\rho \Psi_1$, $\Phi_1 \subseteq \Phi_2$, $\Psi_1 \subseteq \Psi_2$ y sea J una interpretación cualquiera. Para demostrar que $\Phi_2 \vdash_\rho \Psi_2$ consideremos $A_1 = A_J(\Phi_1)$ y $C_1 = A_J(\Psi_1)$, $i=1,2$. Si $A_2 \notin dom(\rho)$ o $A_2 = A_1$, no hay nada que demostrar. Si no, demostremos que para cada C' tal que $A_2 \rho C'$ se tiene que $C_2 \supseteq C'$.

En efecto, si $A_2 \rho C'$ entonces como $\varnothing \subset A_1 \subset A_2$ (recordar que $J(\varnothing) = \{\underline{t}\}$) resultará que, por ser ρ monótono, existe $C \subseteq C'$ tal que $A_1 \rho C$, con lo cual $C_1 \supseteq C$, pues $\Phi_1 \vdash_\rho \Psi_1$, y entonces $C_2 \supseteq C'$, pues $C_2 \supseteq C_1$ y $C' \supseteq C$. ■

El recíproco del lema anterior es falso. Para demostrarlo razonemos como sigue. Consideremos la consecuencia lógica $\vdash_\rho$ inducida por el germen $\rho = \{(w,s), (w,u), (s,s), (u,s), (u,u)\}$. Es inmediato demostrar que $\vdash_\rho$ es monótona, sin embargo ρ no lo es como lo prueba el hecho de que (w,s), $(u,u) \in \rho$. Obsérvese que $u \rho s$, $u \rho u$ siendo $u \neq s$. A continuación vemos que sólo este tipo de comportamiento puede ser causa de que una consecuencia monótona tenga un germen no monótono.

3.-Definición. *Gérmenes deterministas.*

Un germen ρ se dice determinista si y solo si para todo $A \subseteq Bool$ existe a lo más un $C \subseteq Bool$ tal $A \rho C$. ■

Es evidente que todos los gérmenes definidos al final de la sección anterior son deterministas.

A ciertos gérmenes puede asociárseles otros deterministas como demuestra el siguiente resultado:

4.-Lema.

Si ρ es un germen tal que para todo $A \in dom(\rho)$ se tiene:

$$\cap \{C \;/\; A \rho C\} \neq \varnothing$$

entonces existe un germen determinista $\bar{\rho}$ tal que se verifica para cualquier conjunto de fórmulas $\Phi \cup \Psi \cup \{\psi\}$:

(a) $\Phi \vdash_{\bar{\rho}} \Psi \Rightarrow \Phi \vdash_{\rho} \Psi$

(b) $\Phi \vdash_{\rho} \psi \Rightarrow \Phi \vdash_{\bar{\rho}} \psi$.

Demostración. Dado un germen cualquiera ρ en las condiciones del lema, le asociamos la relación $\bar{\rho}$ definida por:

$$\bar{\rho} = \{(A, C) \;/\; A \in dom(\rho),\ C = \cap A\rho\}$$

donde $A\rho = \{C \;/\; A \rho C\}$.

Obsérvese entonces que $\bar{\rho}$ es un germen y, por construcción, es determinista.

(a) $\Phi \vdash_{\bar{\rho}} \Psi \Rightarrow \Phi \vdash_{\rho} \Psi$. Sean una interpretación $\mathfrak{I}$ y un conjunto $C \subseteq Bool$ cualesquiera tales que $A_{\mathfrak{I}}(\Phi) \rho C$. Entonces $A_{\mathfrak{I}}(\Phi) \bar{\rho} D$, para algún $D \subseteq C$. Como $\Phi \vdash_{\rho} \Psi$ entonces $C_{\mathfrak{I}}(\Psi) \ni D$ con lo que $C_{\mathfrak{I}}(\Psi) \ni C$.

(b) $\Phi \vdash_{\rho} \psi \Rightarrow \Phi \vdash_{\bar{\rho}} \psi$. Como en (a) supongamos que $A_{\mathfrak{I}}(\Phi) \bar{\rho} C$ entonces, por construcción, $C = \cap \{D \;/\; A_{\mathfrak{I}}(\Phi) \rho D\}$. Pero dado D tal que $A_{\mathfrak{I}}(\Phi) \rho D$ sabemos que $C_{\mathfrak{I}}(\psi) \in D$, pues $\Phi \vdash_{\rho} \psi$, luego $C_{\mathfrak{I}}(\psi) \in C$. ■

5.-Lema.

Si la consecuencia lógica $\vdash_{\rho}$ inducida por un germen ρ es monótona y ρ es determinista entonces ρ es monótono.

Demostración. Por contraposición, supongamos que ρ no es monótono entonces existen A, A', $C' \subseteq Bool$ tales que $\varnothing \subset A \subset A' \rho C'$ y se da alguna de las siguientes condiciones:

(i) $A \notin dom(\rho)$

(ii) existe C no incluido en C' tal que $A \rho C$.

En el caso (i) elegimos $b \in Bool \setminus C'$, que existe pues $A' \rho C'$ y ρ es un germen, luego $C' \neq Bool$. Entonces $\Phi_A \vdash_\rho \varphi_b$ (recuérdese esta notación introducida en la sección anterior), pues $A \notin dom(\rho)$, $\Phi_A \subset \Phi_{A'}$, y no se tiene $\Phi_{A'} \vdash_\rho \varphi_b$, pues para cualquier J: $A_J(\Phi_{A'}) = A' \rho C'$ (observar que $A_J(\Phi_{A'}) = J(\Phi_{A'})$ pues $A' \neq \emptyset$ ya que $\emptyset \subset A \subset A'$), pero $C_J(\varphi_b) = b \notin C'$. Luego $\vdash_\rho$ no es monótona.

En el caso (ii) elegimos $c \in C \setminus C'$, que existe pues C no está incluido en C', y tenemos $\Phi_A \vdash_\rho \varphi_c$ (pues para cualquier J y cualquier $D \subseteq Bool$, si $A_J(\Phi_A) = A \rho D$ entonces $D = C$, pues ρ es determinista, y se tiene $C_J(\varphi_c) = c \in D$), $\Phi_A \subset \Phi_{A'}$, y no es cierto que $\Phi_{A'} \vdash_\rho \varphi_c$ (pues para cualquier J: $A_J(\Phi_{A'}) = J(\Phi_{A'}) = A' \rho C'$ e $C_J(\varphi_c) = c \notin C'$). Por tanto en este caso $\vdash_\rho$ tampoco es monótona, como queríamos demostrar. ■

6.- GERMENES Y CONSECUENCIAS INDUCIDAS EN LOGICAS 3-VALORADAS.

Nos ocuparemos en esta sección de estudiar los gérmenes que existen para el caso de tres valores booleanos. En adelante, pues, supondremos que $B = \{\underline{t}, \underline{f}, \perp\}$ y que ρ es un subconjunto de $\wp(B) \times \wp(B)$. Impondremos, sin embargo, ciertas restricciones, que formularemos en forma de principios, de manera que las consecuencias lógicas que resulten tengan algunas propiedades deseables.

I. Principio de Contradicción.

Es lógico suponer que las consecuencias lógicas que estudiemos extiendan el comportamiento de la dada en lógica clásica. En este sentido pediremos que un conjunto de fórmulas sea consecuencia de cualquier otro en el que alguna de las hipótesis sea falsa.

1.-Definición.

Decimos que una consecuencia lógica $\vdash$ satisface el principio de contradicción, que abreviamos mediante (CT), si y solo si para todo conjunto de fórmulas $\Phi \cup \Psi$ se tiene $\Phi \cup \{F\} \vdash \Psi$.

Decimos que un germen ρ satisface (CT) si y solo si para todo $A \in dom(\rho)$: $\underline{f} \notin A$. ■

Veamos la equivalencia de los conceptos anteriores en el sentido del siguiente lema:

2.-Lema.

Un germen ρ satisface (CT) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (CT).

Demostración. Supongamos que ρ satisface (CT) entonces, dada una interpretación $\mathfrak{I}$ cualquiera, $A_{\mathfrak{I}}(\Phi \cup \{F\})\rho C$ no se cumple para ningún C, pues $\underline{f} \notin A$, para todo $A \in dom(\rho)$. Por tanto $\Phi \cup \{F\} \vdash_\rho \Psi$ es trivialmente cierto.

Para el recíproco, razonemos por contraposición. Supongamos que ρ no satisface (CT) entonces $A \cup \{\underline{f}\}\rho C$, para ciertos A, C. Como ρ es un germen elijamos $b \in B \setminus C$ y tendremos que no es cierto que $\Phi_A \cup \{F\} \vdash_\rho \varphi_b$, con lo que $\vdash_\rho$ no satisface (CT). ∎

II. Principio de Consistencia.

En lógica clásica no es posible concluir algo contradictorio a partir de ningún conjunto satisfactible de fórmulas. En nuestro caso tenemos:

3.-Definición.

Decimos que una consecuencia lógica $\vdash$ satisface el principio de consistencia, abreviadamente (CN), si y solo si para todo conjunto de fórmulas Φ, si $Sat_{\mathfrak{s}}\ \Phi$ entonces no es el caso que $\Phi \vdash F$.

Decimos que un germen ρ satisface (CN) si y solo si existe C tal que $\underline{f} \notin C$ y $s\rho C$. ∎

4.-Lema.

Un germen ρ satisface (CN) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (CN).

Demostración. Supongamos que ρ es un germen que satisface (CN) y sea Φ un conjunto de fórmulas fuertemente satisfactible. Entonces existe una interpretación $\mathfrak{I}$ tal que $A_{\mathfrak{I}}(\Phi)=s$. Por tanto $A_{\mathfrak{I}}(\Phi)\rho C$ para algún C tal que $\underline{f} \notin C$. Pero entonces no se tiene $\Phi \vdash_\rho F$ con lo que $\vdash_\rho$ satisface (CN).

Supongamos ahora que ρ no satisface (CN) con lo que se tendrá que para todo C: $s\rho C \Rightarrow C \ni \underline{f}$. Entonces $T \vdash_\rho F$ y, de nuevo, $\vdash_\rho$ no satisface (CN). ∎

III. Principio de Tautología.

Como en lógica clásica, supondremos que una conclusión idénticamente cierta es consecuencia del conjunto vacío de premisas.

5.-Definición.

Decimos que una consecuencia lógica $\vdash$ satisface el principio de tautología, abreviadamente (TL), si y solo si se tiene $\vdash T$.

Decimos que un germen ρ satisface (TL) si y solo si para todo C tal que $s\rho C$, se verifica $\underline{t} \in C$. ■

6.-Lema.

Un germen ρ satisface (TL) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (TL).

Demostración. Si ρ satisface (TL) entonces $\underline{t} \in C$ siempre que $s\rho C$ con lo que $\vdash_\rho T$ (recuérdese que, por definición de consecuencia lógica inducida por un germen, $A_J(\varnothing)=s$, para toda interpretación J).

Si ρ no satisface (TL) entonces existe C tal que $s\rho C$ y $\underline{t} \notin C$, con lo que no se tendrá $\vdash_\rho T$ y $\vdash_\rho$ no satisface (TL). ■

Existe otra condición que es coherente pedir que satisfagan los gérmenes, pero que no es reflejo de ninguna situación similar en lógica clásica.

7.-Definición. *Gérmenes superconsistentes.*

Un germen ρ se dice que es superconsistente si y solo si se tiene que para todo $C \in ran(\rho)$: $\underline{f} \notin C$. ■

La idea es que una tal ρ no impone condiciones relativas a la presencia de fórmulas falsas en el consecuente. Obsérvese que (CN) exigía la consistencia bajo la satisfactibilidad fuerte en el sentido de que de un conjunto de fórmulas que tuviera un modelo fuerte no era posible deducir algo falso (impidiendo así que $T \vdash F$). En un germen superconsistente pedimos que esto mismo suceda para cualquier otra satisfactibilidad, débil o indefinida, sobre la cual diga algo la consecuencia lógica inducida.

En adelante estudiaremos aquellos gérmenes que, amén de ser deterministas, monótonos y superconsistentes, satisfacen los tres principios discutidos más arriba.

8.-Definición. *Gérmenes regulares.*

Un germen se dice regular si y solo si es determinista, monótono, superconsistente y satisface los principios de contradicción (CT), consecuencia (CN), y tautología (TL). ■

Una vez definidas las condiciones que debe satisfacer un germen pasemos a estudiar cuáles resultan en el caso 3-valorado.

Un germen regular cualquiera ρ debe cumplir:

(i) Para C=s o C=w (o excluyente): $s\rho C$. En efecto, por consistencia existe C tal que $s\rho C$ y $\underline{f}\notin C$. Por tautología $\underline{t}\in C$ y, por ser ρ un germen, C=s o C=w. Por determinismo, C está unívocamente determinado.

(ii) $dom(\rho)\subseteq\{s, u, w\}$. Por contradicción, ningún $A\notin dom(\rho)$ puede cumplir $A\supset\underline{f}$.

Obsérvese además que, por monotonía, si $w\notin dom(\rho)$ entonces $u\notin dom(\rho)$.

(iii) $ran(\rho)\subseteq\{s, u, w\}$. Por ser ρ un germen, ningún $C\in ran(\rho)$ puede ser $\emptyset$ o B. Por superconsistencia, ningún $C\in ran(\rho)$ puede cumplir $\underline{f}\in C$.

Obsérvese que si $w\rho C$ entonces $C\neq u$ pues en otro caso $s\rho u$, por monotonía, contra el principio de tautología. Luego $C\in\{s, w\}$.

Considerando (i)-(iii) y el carácter monótono y determinista de los gérmenes, deducimos fácilmente cuáles resultan, de entre los regulares. Para ello distinguimos los siguientes casos, observando que, según (i), $s\notin dom(\rho)$.

Caso 1: $dom(\rho) = \{s\}$. Por la condición (i), ρ sólo puede ser $\{(s, s)\}$, es decir la ss-consecuencia (cf. sección 2), o $\{(s, w)\}$, esto es la sw-consecuencia.

Si $dom(\rho) \supset \{s\}$ entonces, por monotonía, solamente son posibles otros dos casos:

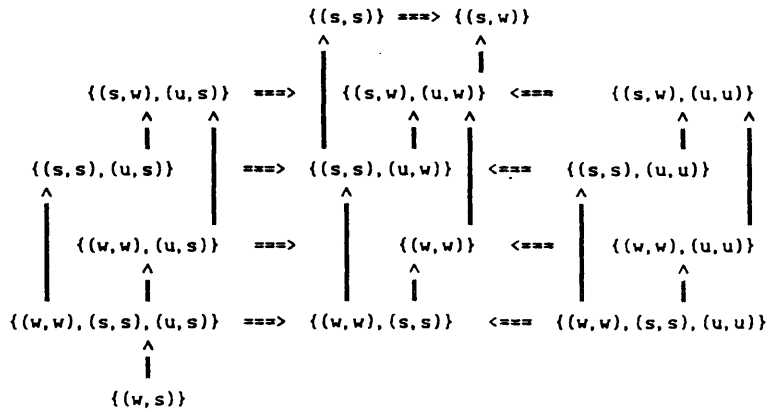
Caso 2: $dom(\rho) = \{s, u\}$. Por (i) tendremos que $s\rho C_s$, siendo $C_s=s$ o $C_s=w$. Por otro lado tendremos que $u\rho C'_u$ para un único C'_u que, por (iii), verifica $C'_u\in\{s, u, w\}$. Por tanto, ρ es de una de las seis posibles formas determinadas por $\{(s, b), (u, b')\}$, con $b\in\{s, w\}$ y $b'\in\{s, u, w\}$.

Caso 3: $dom(\rho) = \{s, u, w\}$. Por (i) tendremos que $s\rho C_s$ siendo $C_s=s$ o $C_s=w$. Por otro lado tendremos C_u, C_w unívocamente determinados tales que $u\rho C_u$, $w\rho C_w$. Por monotonía, deberá verificarse que C_s, $C_u \subseteq C_w$, con, según (iii), $C_w\in\{s, w\}$. Si $C_w=s$ entonces $C_s=C_u=s$, por monotonía. Escribiremos, entonces, simplemente $\{(w, s)\}$, es decir la ws-consecuencia. Si $C_w=w$ entonces ρ es de una de las seis posibles formas determinadas por $\{(s, b), (u, b'),$

(w, w)}, con b∈{s, w} y b'∈{s, u, w}. Para b=w y b'=w se obtiene, abreviando como antes, {(w, w)}, esto es la ww-consecuencia; para b=s y b'=w escribimos {(s, s), (w, w)}, que es la consecuencia de Blamey (cf. sección 3).

Es fácil demostrar que los quince gérmenes que acabamos de obtener son regulares.

Las relaciones de inclusión que existen entre las quince consecuencias lógicas inducidas por estos gérmenes se expresan en el siguiente diagrama donde, para abreviar la escritura, se supone que ρ ===> ρ' representa $\vdash_\rho \subseteq \vdash_{\rho'}$.



Las inclusiones que muestra este diagrama son propias, es decir, dados ρ y ρ' que aparezcan en dos nodos diferentes del diagrama tales que ρ ===> ρ', se tiene que $\vdash_\rho \subset \vdash_{\rho'}$. En efecto, si ρ ===> ρ' entonces pueden darse los siguientes casos:

1) existen dos pares $(a,b)\in\rho$ y $(a,b')\in\rho'$ tales que $b<b'(=w)$. Si b=s entonces $\varphi_a \vdash_\rho \Omega$, pero no $\varphi_a \vdash_{\rho'} \Omega$. Si b=u entonces $\varphi_a \vdash_{\rho'} T$, pero no $\varphi_a \vdash_\rho T$.

2) existe a tal que $a\in dom(\rho)\setminus dom(\rho')$ entonces a=w o a=u (observar que el paso de ρ a ρ' corresponde entonces al paso del caso 3 al 2 o del 2 al 1). Si a=w se tendrá $\Omega, T \vdash_{\rho'} F$, pero no, por superconsistencia, para $\vdash_\rho$. Si a=u entonces $\Omega \vdash_{\rho'} F$, pero no, análogamente, para $\vdash_\rho$.

Concluimos esta sección haciendo notar que dos hechos se mantienen igual que en la definición 2.1: {(w,s)} induce la consecuencia lógica más fuerte, y {(s,w)}, la más débil.

7.- CARACTERISTICAS GENERALES DE LAS CONSECUENCIAS LOGICAS REGULARES.

El propósito de esta sección es caracterizar las consecuencias lógicas que acabamos de definir con el fin de encontrar criterios que nos permitan optar adecuadamente por una de ellas. Para tal propósito, estudiaremos su comportamiento frente a cuatro propiedades características de la lógica clásica: la reflexividad, el *modus ponens*, la regla de deducción y la regla de corte.

En adelante supondremos pues que ρ es siempre un germen regular.

(A) Reflexividad.

1.-Definición.

Decimos que una consecuencia lógica $\vdash$ es reflexiva si y solo si para todo conjunto de fórmulas $\Phi \cup \Psi$ se tiene que $\Phi \vdash \Psi$ siempre que $\Phi \cap \Psi \neq \emptyset$.

Decimos que un germen ρ es reflexivo si y solo si para todo A, C se tiene: $A \rho C \Rightarrow A \subseteq C$. ■

2.-Lema.

Un germen ρ es reflexivo si y solo si la consecuencia lógica $\vdash_\rho$ que induce es reflexiva.

Demostración. Sea ρ reflexivo y supongamos que $\Phi \cap \Psi \neq \emptyset$. Sea $\mathfrak{I}$ una interpretación cualquiera tal que $A_{\mathfrak{I}}(\Phi) \rho C$. Como ρ es reflexivo se tendrá que $\mathfrak{I}(\Phi) \subseteq C$. Entonces $C_{\mathfrak{I}}(\Psi) \ni C$, pues $\Phi \cap \Psi \neq \emptyset$. Por tanto $\Phi \vdash_\rho \Psi$.

Recíprocamente, por contraposición, supongamos que $A \rho C$ y que A no está contenido en C. Elijamos $a \in A \setminus C$ con lo que $\varphi_a \in \Phi_A$, pero no es el caso que $\Phi_A \vdash_\rho \varphi_a$. ■

De 2 se deduce que los cuatro gérmenes que contienen el par (u, s) y el germen {(w, s)}, no son reflexivos y, por tanto, las consecuencias lógicas que inducen, tampoco.

(B) *Modus ponens*.

3.-Definición.

Decimos que una relación de consecuencia lógica $\vdash$ satisface el *modus ponens*, abreviadamente MP, si y solo si para todo par de fórmulas φ, ψ

se tiene $\varphi,\ \varphi\rightarrow\psi \vdash \psi$.

Decimos que un germen ρ satisface (MP) si y solo si $u\notin dom(\rho)$. ■

4.-Lema.

Un germen ρ satisface (MP) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (MP).

Demostración. Supongamos que ρ satisface (MP) y que existe una interpretación $\mathfrak{I}$ tal que $A_{\mathfrak{I}}(\Phi)=\mathfrak{I}(\Phi)\rho C$, $\Phi=\{\varphi,\ \varphi\rightarrow\psi\}$, e $\mathfrak{I}(\psi)\notin C$. Si $\mathfrak{I}(\varphi)=\perp$ entonces $u\subseteq\mathfrak{I}(\Phi)\rho C$ con lo que, por monotonía, supondría que $u\in dom(\rho)$, contra (MP). Si $\mathfrak{I}(\varphi)=\underline{t}$ entonces $\mathfrak{I}(\psi)\neq\underline{f}$ (pues en otro caso $\underline{f}\in\mathfrak{I}(\Phi)\rho C$, contra (CT)); similarmente se razona que $\mathfrak{I}(\psi)\neq\perp$, para no contradecir (MP) pues entonces $\mathfrak{I}(\varphi\rightarrow\psi)=\perp$; por tanto $\mathfrak{I}(\psi)=\underline{t}$. Pero entonces $\mathfrak{I}(\Phi)=s\rho C$ y $\underline{t}\notin C$, contra (TL). Luego $\mathfrak{I}(\varphi)=\underline{f}$ con lo que, como antes, $\underline{f}\in\mathfrak{I}(\Phi)$, contradiciendo (CT).

Para el recíproco, razonemos por contraposición. Si ρ no satisface (MP) entonces $u\in dom(\rho)$ luego $u\rho C$ con $\underline{f}\notin C$, por superconsistencia. Entonces no es el caso que $\{\Omega,\ \Omega\rightarrow F\} \vdash_\rho F$, con lo que $\vdash_\rho$ no satisface (MP). ■

El contraejemplo anterior muestra cómo obtener una versión más débil de (MP). Para ello exigimos que el antecedente esté definido, es decir:

$$\varphi,\ \Delta\varphi,\ \varphi\rightarrow\psi \vdash \psi$$

Escribiremos (MP_Δ) para referirnos a ella.

5.-Definición.

Decimos que una relación de consecuencia $\vdash$ satisface (MP_Δ) si y solo si para todo par de fórmulas $\varphi,\ \psi$ se tiene que $\varphi,\ \Delta\varphi,\ \varphi\rightarrow\psi \vdash \psi$.

Decimos que un germen ρ satisface (MP_Δ) si y solo si $(w,s)\notin\rho$. ■

6.-Lema.

Si un germen ρ satisface (MP) entonces satisface (MP_Δ).

Demostración. En efecto, en otro caso $w\in dom(\rho)$ con lo que, por monotonía, se tendría que $u\in dom(\rho)$, contra (MP). ■

7.-Lema.

Un germen ρ satisface (MP_Δ) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (MP_Δ).

Demostración. Supongamos que ρ satisface (MP_Δ) y que existe una interpretación $\mathfrak{I}$ tal que $A_{\mathfrak{I}}(\Phi)=\mathfrak{I}(\Phi)\rho C$, $\Phi=\{\varphi,\ \Delta\varphi,\ \varphi\rightarrow\psi\}$, e $\mathfrak{I}(\psi)\notin C$. Por (CT),

$\mathfrak{I}(\varphi)$, $\mathfrak{I}(\Delta\varphi)\neq\underline{f}$ luego $\mathfrak{I}(\Delta\varphi)=\underline{t}$ y, entonces, $\mathfrak{I}(\varphi)=\underline{t}$ luego $\mathfrak{I}(\psi)\neq\underline{f}$, para no contradecir (CT) (pues en otro caso $\mathfrak{I}(\varphi\rightarrow\psi)=\underline{f}$); si $\mathfrak{I}(\psi)=\perp$ entonces $\mathfrak{I}(\Phi)=w\rho C$; pero $C\in\{s, w\}$ luego, como ρ satisface (MP_Δ), $C\neq s$ y entonces $\mathfrak{I}(\psi)=\perp\in C=w$, contra lo supuesto. Por tanto $\mathfrak{I}(\psi)=\underline{t}$ luego $\mathfrak{I}(\Phi)=s\rho C$ e $\mathfrak{I}(\psi)=\underline{t}\notin C$, contra (TL).

Para el recíproco, razonando por contraposición, tenemos que si $(w,s)\in\rho$ entonces no es el caso que $\{T, \Delta T, T\rightarrow\Omega\} \vdash_\rho \Omega$, con lo que $\vdash_\rho$ no satisface (MP_Δ). ■

(C) Regla de deducción.

8.-Definición.

Decimos que una relación de consecuencia lógica $\vdash$ satisface la regla de deducción, abreviadamente (RD), si y solo si para todo conjunto de fórmulas $\Phi \cup \{\varphi, \psi\}$ se tiene: $\Phi, \varphi \vdash \psi \Rightarrow \Phi \vdash \varphi\rightarrow\psi$.

Decimos que un germen ρ satisface (RD) si y solo si ρ es $\{(w, w)\}$, $\{(w, s)\}$ o $\{(s, w)\}$. ■

9.-Lema.

Un germen ρ satisface (RD) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (RD).

Demostración. Supongamos que ρ satisface (RD) y distingamos casos según la ab-consecuencia que induzca. Para la aw-consecuencia, $a\in\{s, w\}$, razonamos como sigue. Supongamos que $\Phi, \varphi \vdash_\rho \psi$ y sea $\mathfrak{I}$ una interpretación cualquiera tal que $A_{\mathfrak{I}}(\Phi)\rho w$; si $\mathfrak{I}(\varphi\rightarrow\psi)\in w$ entonces $\mathfrak{I}(\varphi\rightarrow\psi)=\underline{f}$ con lo que $\mathfrak{I}(\varphi)=\underline{t}$ e $\mathfrak{I}(\psi)=\underline{f}$, luego $A_{\mathfrak{I}}(\Phi \cup \{\varphi\})=\mathfrak{I}(\Phi \cup \{\varphi\})\rho w$ (pues o bien $A_{\mathfrak{I}}(\Phi)\in\{s, w\}$ y se tendrá $\mathfrak{I}(\Phi\cup\{\varphi\})$ $=A_{\mathfrak{I}}(\Phi)\in\{s, w\}$, o bien $A_{\mathfrak{I}}(\Phi)=u$ y como $A_{\mathfrak{I}}(\Phi)\rho w$, estaremos en el caso de la ww-consecuencia y tendremos también que $\mathfrak{I}(\Phi \cup \{\varphi\})=w\rho w$); por ser $\Phi, \varphi \vdash_\rho \psi$, resultará que $\mathfrak{I}(\psi)\in w$, de manera que si $\mathfrak{I}(\psi)=\underline{f}$, contradecimos la superconsistencia. El caso de la ws-consecuencia es similar. Si, como antes, $A_{\mathfrak{I}}(\Phi)\rho s$ e $\mathfrak{I}(\varphi\rightarrow\psi)\neq\underline{t}$ entonces $\mathfrak{I}(\varphi)\neq\underline{f}$ con lo que $\mathfrak{I}$ sigue siendo modelo débil de $\Phi \cup \{\varphi\}$; como $\Phi, \varphi \vdash_\rho \psi$ entonces $\mathfrak{I}(\Phi \cup \{\varphi\})\rho s$ y por tanto $\mathfrak{I}(\psi)=\underline{t}$, con lo que $\mathfrak{I}(\varphi\rightarrow\psi)=\underline{t}$, contra lo que habíamos supuesto.

Si ρ no satisface (RD) entonces razonamos según la siguiente cadena de casos. Supongamos que $w\notin dom(\rho)$. Si $u\notin dom(\rho)$ entonces ρ es $\{(s, s)\}$ que verifica $\Omega \vdash_\rho \Omega$, pero no $\vdash_\rho \Omega\rightarrow\Omega$. Si $(u, b)\in\rho$, para algún $b\in\{s, w, u\}$, entonces $\Omega, T \vdash_\rho F$, pero no $\Omega \vdash_\rho T\rightarrow F$. Supongamos entonces que $(w, a)\in\rho$, para cierto $a\in\{s,w\}$. Si ρ no induce la ws-consecuencia entonces $a=w$. Si $(s, s)\in\rho$,

vale el mismo contraejemplo que para $\{(s, s)\}$. Si no, $(s, w)\in\rho$ y, como $u\in dom(\rho)$, por monotonía, caben tres posibilidades:

(i) $(u, w)\in\rho$ resultando que $\rho=\{(w, w)\}$ que, según hemos visto, satisface (RD)

(ii) $(u, s)\in\rho$. Entonces $\Omega, T \vdash_\rho \Omega$, pero no $\Omega \vdash_\rho T\rightarrow\Omega$

(iii) $(u, u)\in\rho$. Entonces $\Omega, T \vdash_\rho T$, pero no $\Omega \vdash_\rho T\rightarrow T$. ■

Como se hizo más arriba, es posible encontrar una versión más débil (RD_Δ) de (RD) imponiendo de nuevo que el antedente de la implicación esté definido; es decir: $\Phi, \varphi \vdash \psi \Rightarrow \Phi, \Delta\varphi \vdash \varphi\rightarrow\psi$. Por monotonía es inmediato demostrar que todo germen regular que satisface (RD), satisface (RD_Δ).

10.-<u>Lema</u>.

Todas las consecuencias lógicas inducidas por gérmenes regulares satisfacen RD_Δ.

<u>Demostración</u>. Supongamos que $\Phi, \varphi \vdash_\rho \psi$ y sea $\mathfrak{I}$ una interpretación cualquiera tal que $A_{\mathfrak{I}}(\Phi \cup \{\Delta\varphi\})=\mathfrak{I}(\Phi \cup \{\Delta\varphi\})\rho C$, entonces $\mathfrak{I}(\Phi \cup \{\Delta\varphi\})\in\{s, w\}$, pues $\mathfrak{I}(\Delta\varphi)\neq\perp$, y $C\in\{s, w\}$. Por tanto, $\mathfrak{I}(\Delta\varphi)=\underline{t}$ con lo que o bien $\mathfrak{I}(\varphi)=\underline{f}$, y se tendrá que $\mathfrak{I}(\varphi\rightarrow\psi)=\underline{t}$ y, por (TL), $\mathfrak{I}(\varphi\rightarrow\psi)\in C$, o bien $\mathfrak{I}(\varphi)=\underline{t}$, y resultará que $\mathfrak{I}(\Phi \cup \{\Delta\varphi\})=\mathfrak{I}(\Phi \cup \{\varphi\})$ con lo que, por ser $\Phi, \varphi \vdash_\rho \psi$, habrá de tenerse $\mathfrak{I}(\varphi\rightarrow\psi)=\mathfrak{I}(\psi)\in C$. ■

(D) Regla de corte.

11.-<u>Definición</u>.

Decimos que una relación de consecuencia lógica $\vdash$ satisface la regla de corte (o transitividad), abreviadamente (RC), si y solo si para todo conjunto de fórmulas $\Phi \cup \Phi' \cup \Psi \cup \Psi' \cup \{\eta\}$ se tiene:

$$\Phi \vdash \{\eta\} \cup \Psi,\ \Phi' \cup \{\eta\} \vdash \Psi' \Rightarrow \Phi \cup \Phi' \vdash \Psi \cup \Psi'.$$

Decimos que un germen ρ satisface (RC) si y solo si ρ es o bien $\{(w, s)\}$ o $\{(w, w), (s, s), (u, u)\}$ o un subconjunto de éste que sea un germen regular. ■

En caso de que una consecuencia lógica no satisfaga (RC) entonces no es posible encontrar para ella cálculos correctos y completos "al estilo de Hilbert". En su lugar será necesario la utilización de cálculos de secuencias, como veremos en el capítulo siguiente.

12.-Lema.

Un germen ρ satisface (RC) si y solo si la consecuencia lógica que induce $\vdash_\rho$ satisface (RC).

Demostración. Supongamos que ρ es un germen distinto de $\{(w, s)\}$, que satisface (RC) (observar que, entonces, ρ es un conjunto de pares de la forma (a, a), para a recorriendo algún subconjunto no vacío de $\{s, w, u\}$) y que se tiene $\Phi \vdash_\rho \{\eta\} \cup \Psi$ y $\Phi' \cup \{\eta\} \vdash_\rho \Psi'$. Sea $\mathfrak{I}$ una interpretación cualquiera tal que $A_{\mathfrak{I}}(\Phi \cup \Phi')\rho C$ y $A_{\mathfrak{I}}(\Phi \cup \Phi')=a$, $a\in\{s, w, u\}$, con lo que $C=a$, por lo que acabamos de decir sobre ρ. Si no es el caso que tengamos $C\ni C_{\mathfrak{I}}(\Psi \cup \Psi')$ entonces, como $\Phi \vdash_\rho \{\eta\} \cup \Psi$, tendremos que $\mathfrak{I}(\eta)\in a$. Pero $\Phi' \cup \{\eta\} \vdash_\rho \Psi'$ luego $C\ni C_{\mathfrak{I}}(\Psi')$ y, entonces, $C\ni C_{\mathfrak{I}}(\Psi \cup \Psi')$, contra lo supuesto. Si ρ es $\{(w, s)\}$ entonces dada una interpretación $\mathfrak{I}$ cualquiera tal que $A_{\mathfrak{I}}(\Phi \cup \Phi')=w\rho s$ e $\mathfrak{I}$ no sea modelo fuerte de ninguna fórmula de $\Psi \cup \Psi'$ entonces, como $\Phi \vdash_\rho \{\eta\} \cup \Psi$, se tendrá que $\mathfrak{I}(\eta)=\underline{t}$, luego $A_{\mathfrak{I}}(\Phi' \cup \{\eta\})=w$, con lo que al ser $\Phi' \cup \{\eta\} \vdash_\rho \Psi'$, tendremos que $\mathfrak{I}$ será modelo fuerte de alguna fórmula de Ψ', contra lo supuesto.

Recíprocamente, por contraposición, si ρ no satisface (RC) entonces razonamos como sigue. Empecemos suponiendo que $(s, w)\in\rho$. Si $w\notin dom(\rho)$ tendremos que $T \vdash_\rho \Omega$ y $\Omega, T \vdash_\rho F$, pero no $T \vdash_\rho F$. Si no es el caso anterior entonces $(w, w)\in\rho$ y $(u, b)\in\rho$, para algún $b\in\{s, w, u\}$, por monotonía. Supongamos ahora que $(u, s)\in\rho$; tendremos entonces $\Omega \vdash_\rho T$, Ω y $\Omega, T \vdash_\rho \Omega$, pero no $\Omega \vdash_\rho \Omega$. Si, en cambio, $(u, s)\notin\rho$ tendremos que o bien $(u, w)\in\rho$ y ρ será $\{(w, w)\}$, que acabamos de ver satisface (RC), o bien $(u, u)\in\rho$ y ρ será el germen $\{(w, w), (u, u)\}$, que también satisface (RC). Finalmente, si $(s, w)\notin\rho$ entonces $(s, s)\in\rho$ y volvemos a distinguir según $(w, s)\in\rho$ (teniéndose entonces que ρ es $\{(w, s)\}$ que satisface (RC)) o no pertenezca. En este caso, si se tiene que $(w, w)\in\rho$ y ρ no satisface (RC) entonces $(u, s)\in\rho$ y vale el último contraejemplo para demostrar que $\vdash_\rho$ no es transitiva; si, por el contrario, $(w, w)\notin\rho$ entonces para el caso $(u, s)\in\rho$ volvemos a tomar éste contraejemplo, para $(u, w)\in\rho$ observamos que $\Omega \vdash_\rho T$ y $\Omega, T \vdash_\rho F$, pero no $\Omega \vdash_\rho F$ y para el resto de los casos ($(u, u)\in\rho$ o $u\notin dom(\rho)$) hacemos notar que resultan gérmenes ($\{(s, s), (u, u)\}$ y $\{(s, s)\}$, respectivamente) que satisfacen (RC). ■

Como en (B) y (C), existe una versión más débil RC_Δ de la regla de corte que consiste en: ($\Phi \vdash \Psi$, $\varphi\wedge\Delta\varphi$ y Φ', $\varphi \vdash \Psi' \Rightarrow \Phi$, $\Phi' \vdash \Psi$, Ψ'). En primer lugar demostremos que realmente RC_Δ es una versión más débil de RC.

13.-Lema.

Si $\vdash_\rho$ satisface RC entonces satisface RC_Δ.

Demostración. Evidentemente, basta ver que ρ satisface:

$$\Phi \vdash_\rho \Psi,\ \varphi\wedge\Delta\varphi \Rightarrow \Phi \vdash_\rho \Psi,\ \varphi$$

Supongamos que $\Phi \vdash_\rho \Psi,\ \varphi\wedge\Delta\varphi$ y sea $\mathfrak{I}$ una interpretación cualquiera tal que $A_{\mathfrak{I}}(\Phi)\rho C$ entonces $C\ni C_{\mathfrak{I}}(\Psi \cup \{\varphi\wedge\Delta\varphi\})$. Si $C\in C_{\mathfrak{I}}(\Psi)$ hemos acabado; si no, por ser ρ superconsistente, $\mathfrak{I}(\varphi\wedge\Delta\varphi)\neq\underline{f}$ luego $\mathfrak{I}(\varphi\wedge\Delta\varphi)=\underline{t}$ y, por tanto, $\mathfrak{I}(\varphi)=\underline{t}$. Si se tuviera $C_{\mathfrak{I}}(\Psi \cup \{\varphi\}) \cap C = \varnothing$ entonces $\mathfrak{I}(\varphi\wedge\Delta\varphi)$ $(=\mathfrak{I}(\varphi))\in C$, pues $\Phi \vdash_\rho \Psi,\ \varphi\wedge\Delta\varphi$ y $A_{\mathfrak{I}}(\Phi)\rho C$, contra lo supuesto. ■

14.-Lema.

La consecuencia lógica inducida por un germen ρ satisface (RC_Δ) si y solo si ρ es $\{(s, w)\}$, $\{(s, w), (u, u)\}$ o de alguna de las formas de los gérmenes que satisfacen (RC).

Demostración. Si ρ es $\{(s, w)\}$ entonces, supuesto que $\Phi \vdash_\rho \Psi,\ \varphi\wedge\Delta\varphi$ y que $\Phi',\ \varphi \vdash_\rho \Psi'$, si existe una interpretación $\mathfrak{I}$ tal que $A_{\mathfrak{I}}(\Phi \cup \Phi')=s$ y $C_{\mathfrak{I}}(\Psi \cup \Psi')=\{\underline{f}\}$, resultará que, por ser $\Phi \vdash_\rho \Psi,\ \varphi\wedge\Delta\varphi$, entonces $\mathfrak{I}(\varphi\wedge\Delta\varphi)\neq\underline{f}$ luego $\mathfrak{I}(\varphi\wedge\Delta\varphi)=\underline{t}$ luego $\mathfrak{I}(\varphi)=\underline{t}$ y, al ser $\Phi',\ \varphi \vdash_\rho \Psi'$, tendremos que $w\ni C_{\mathfrak{I}}(\Psi')$, contra lo supuesto.

Para $\{(s, w), (u, u)\}$ se razona como antes teniendo en cuenta que si $\mathfrak{I}$ es una interpretación tal que $A_{\mathfrak{I}}(\Phi \cup \Phi')=u$, pero $C_{\mathfrak{I}}(\Psi \cup \Psi') \cap u = \varnothing$ entonces, como por hipótesis $\Phi \vdash_\rho \Psi,\ \varphi\wedge\Delta\varphi$, resultaría que $\mathfrak{I}(\varphi\wedge\Delta\varphi)=\iota$, lo que es absurdo. Quitado este posible caso, el otro que resulta es el mismo que el de $\{(s, w)\}$, que se razona como antes.

Para los casos en que ρ satisface (RC) aplicamos el lema anterior.

Supongamos ahora que ρ no satisface (RC_Δ). Siguiendo el razonamiento del lema 12, si $(s, w)\in\rho$ y $w\notin dom(\rho)$, como ρ no satisface (RC_Δ), entonces o bien $(u, s)\in\rho$ teniéndose $\Omega \vdash_\rho T\wedge\Delta T,\ \Omega$ y $\Omega,\ T \vdash_\rho \Omega$, pero no $\Omega \vdash_\rho \Omega$, o bien $(u, w)\in\rho$, resultando $\Omega \vdash_\rho T\wedge\Delta T$ y $\Omega,\ T \vdash_\rho F$, pero no $\Omega \vdash_\rho F$. Si por el contrario, $(s, w)\notin\rho$ razonamos igual que en el lema 12, tomando ahora los dos contraejemplos, allí usados, en su forma debilitada, que es la que acabamos de dar. ■

Concluimos así el estudio mencionado al principio de esta sección. Resumimos las propiedades que satisfacen o no cada uno de los gérmenes regulares obtenidos, mediante el siguiente cuadro:

	(Ref.)	(MP)	(MP_Δ)	(RD)	(RD_Δ)	(RC)	(RC_Δ)
{(s, w)}	SI	SI	SI	SI	SI	NO	SI
{(s, s)}	SI	SI	SI	NO	SI	SI	SI
{(s, w), (u, u)}	SI	NO	SI	NO	SI	NO	SI
{(s, w), (u, w)}	SI	NO	SI	NO	SI	NO	NO
{(s, w), (u, s)}	NO	NO	SI	NO	SI	NO	NO
{(s, s), (u, u)}	SI	NO	SI	NO	SI	SI	SI
{(s, s), (u, w)}	SI	NO	SI	NO	SI	NO	NO
{(s, s), (u, s)}	NO	NO	SI	NO	SI	NO	NO
{(w, w), (u, u)}	SI	NO	SI	NO	SI	SI	SI
{(w, w)}	SI	NO	SI	SI	SI	SI	SI
{(w, w), (u, s)}	NO	NO	SI	NO	SI	NO	NO
{(w, w), (s, s), (u, u)}	SI	NO	SI	NO	SI	SI	SI
{(w, w), (s, s)}	SI	NO	SI	NO	SI	SI	SI
{(w, w), (s, s), (u, s)}	NO	NO	SI	NO	SI	NO	NO
{(w, s)}	NO	NO	NO	SI	SI	SI	SI

De este cuadro se deduce que sólo algunas de las ab-consecuencias satisfacen el *modus ponens* y la regla de deducción. La regla de corte la satisfacen, además de algunas de ellas, los gérmenes {(s,s), (w,w), (u,u)}, {(w, w), (u, u)}, {(s,s), (u,u)} y la consecuencia de Blamey {(s,s), (w,w)}. Unicamente {(s,w)} verifica a la vez MP y RD y, aunque no cumple la regla de corte, admitirá, según veremos, cálculos de secuencias correctos y completos.

De todo lo dicho deducimos que las relaciones de consecuencia lógica más naturales, en el sentido de que satisfacen más propiedades de la lógica clásica, son las ab-consecuencias y alguna combinación (intersección) de ellas. Es por tanto lógico que optemos por una de éstas para estudiar, en adelante, sistemas formales basados en tres valores booleanos. Por lo que acabamos de decir, elegiremos la más débil de tal modo que en capítulos posteriores escribiremos simplemente $\vdash$ en lugar de $\vdash_{sw}$. En todo caso podemos relacionar cualquier otra con ésta, usando los lemas 2.3 y 2.4.

De los apartados A y B del presente capítulo obtenemos como conclusión los siguientes hechos que serán usados en capítulos posteriores:

(1) Se dispone de un conjunto de tres valores booleanos {$\underline{t}$, $\imath$, $\underline{f}$}, correspondientes a los valores de verdad *cierto*, *indefinido* y *falso*, respectivamente.

(2) Se dispone de un conjunto CK de conectivas, funcionalmente completo, formado por la negación y la disyunción de Kleene, el operador de definición Δ y el operador 0-ádico constantemente indefinido Ω.

(3) La semántica de los cuantificadores existencial y universal es la de Kleene y extiende la disyunción y la conjunción de éste, respectivamente.

(4) La relación de consecuencia lógica es la sw-consecuencia. Con ella se exige que la conclusión no sea falsa si las hipótesis son ciertas.

CAPITULO 2
UNA LOGICA DE PRIMER ORDEN PARA FUNCIONES PARCIALES

El problema fundamental que introducen las funciones parciales en los sistemas formales que las incluyen es el tratamiento de la divergencia. Dicho tratamiento se ha llevado a cabo, en muchos casos, desde un punto de vista estrictamente matemático. Para ello se optaba o bien por transformar las funciones en predicados, considerando su grafo, o bien por asignar valores arbitrarios en los argumentos en que divergen.

Así consideradas, ambas soluciones no son totalmente idóneas para nuestros objetivos. La primera se vuelve inoperante si nos proponemos también el estudio de los predicados parciales, ya que no hacemos más que trasladar el problema de la divergencia de las funciones a la de los predicados. La segunda no es útil desde un punto de vista computacional pues, si bien de las funciones manejadas en matemáticas se supone conocido el dominio de definición, no es éste el caso si nos referimos a algoritmos, para los cuales averiguar si terminan sobre una entrada determinada se convierte en un problema, según se sabe, indecidible.

La solución que se toma entonces es, por una parte, la más complicada, pero, por otra, la más satisfactoria para un estudio del razonamiento formal sobre las funciones computables.

Se trata de introducir un valor arbitrario único para expresar la divergencia de una función y utilizar un mecanismo formal que nos permita averiguar cuándo una función no está definida en un argumento.

Lo primero lo hacemos de la manera obvia: todo conjunto sobre el que toma valores una función parcial f se amplia con un elemento nuevo u de tal forma que f se vuelve total haciendo que valga u en los argumentos en que diverge. Lo segundo lo hacemos mediante el operador de definición Δ del que hablamos en el capítulo anterior, extendiendo su aplicación a los términos. Dicho operador permite el razonamiento formal sobre funciones parciales de dominio no determinado, de la siguiente manera: dada, pongamos por caso, la función parcial f, decir que el valor, si existe, de f en x, para cualquier valor de x, es un número par, se puede expresar mediante la fórmula siguiente $\forall x(\Delta f(x) \rightarrow Par(f(x)))$.

A los predicados parciales se llega si pensamos en el valor semántico que tomaría una fórmula como x/x>0 cuando se valora x como 0. En este caso interpretamos la divergencia de la división diciendo que el valor veritativo de la fórmula no está definido, introduciendo para ello el conjunto de tres valores booleanos mencionado en el capítulo anterior. Obsérvese además que esto supone tratar de manera uniforme tanto a funciones como a predicados parciales pues en ambos casos se reserva un elemento para expresar lo divergente.

Iniciamos pues nuestro estudio bajo estas consideraciones y lo empezamos tratando de obtener un sistema formal básico, una lógica de primer orden. En este marco no será posible, como es bien sabido, un análisis de todas las funciones computables, sin embargo se mostrará la forma que ha de tener cualquier lógica que lo extienda y que aspire a un estudio formal de las funciones recursivas parciales.

El desarrollo de este capítulo se inicia con la definición del lenguaje que emplearemos y su semántica. La sintaxis de los términos será la de primer orden, más dos construcciones al estilo de los lenguajes de programación, "if-then" e "if-then-else", que servirán, entre otras cosas, para definir fórmulas siempre indefinidas. La sintaxis de las fórmulas incluirá, como ya hemos dicho, el operador de definición Δ aplicado tanto a términos como a fórmulas.

La semántica tratará las funciones parciales mediante órdenes parciales completos (cpo's) planos, utilizando su elemento mínimo para representar la divergencia. Ya que el conjunto de tres valores booleanos que emplearemos es también un cpo plano, trataremos de la misma forma, como se ha dicho, a funciones y a predicados.

En la sección segunda presentaremos algunos resultados técnicos, como los conocidos sobre coincidencia y sustitución para lógica de primer orden, aplicados a nuestra lógica.

En la sección tercera introducimos un método de tableaux capaz de tratar funciones parciales y tres valores veritativos. El método de los tableaux es un procedimiento potente para estudiar la teoría de prueba de los sistemas formales de primer orden y nosotros veremos cómo aplicarlo al nuestro.

El método se basa en la construcción de árboles para conjuntos de fórmulas de forma que según sea el árbol, podamos extraer información sobre la satisfactibilidad del conjunto. Los nodos del árbol están etiquetados por fórmulas del lenguaje y según sea la estructura de éstas, el árbol se

extiende de una forma u otra.

En la siguiente sección continuamos la definición de los tableaux para nuestro método, precisando cuándo se consideran cerrados. De los tableaux cerrados se puede concluir la insatisfactibilidad de un conjunto de fórmulas; más exactamente, el método de los tableaux puede utilizarse como un procedimiento de demostración por refutación, usando que un tableau para un conjunto de fórmulas que contenga una contradicción semántica obvia en todas sus ramas hace insatisfactible fuertemente el conjunto de partida.

Demostraremos formalmente esto, la así llamada corrección del método, en la sección quinta y en la siguiente veremos que también es cierto el recíproco, la completitud del método, es decir, si un conjunto de fórmulas no es fuertemente satisfactible, podemos construir para él un tableau tal que todas sus ramas contienen una contradicción semántica obvia. Para ello seguiremos una construcción de modelo (fuerte) a partir de constantes, empleando un concepto adecuado de conjunto de Hintikka, y mostraremos cómo se obtienen dichos conjuntos en la construcción de ciertos tableaux, los que llamamos canónicos.

La sección séptima muestra cómo relacionar el método de los tableaux con la obtención de cálculos de secuencias completos para nuestra lógica, obteniendo condiciones de manera que todo cálculo que las satisfaga sea completo. La siguiente sección presenta un cálculo de secuencias para nuestra lógica y en las secciones décima y undécima se demuestra su corrección y completitud, ésta última apoyada en la relación entre tableaux y cálculo que acabamos de mencionar. Por fin, el capítulo termina con una reseña de los trabajos más importantes que conocemos sobre el tema.

1.- LENGUAJE DE UNA LOGICA DE PRIMER ORDEN PARA FUNCIONES PARCIALES: LFP.

La sintaxis del lenguaje que presentamos es similar a la de primer orden, con la salvedad de dos nuevos tipos de construcciones para términos y para fórmulas.

En adelante supondremos que Σ es una signatura de primer orden con igualdad, es decir, un conjunto (numerable o no) formado por constantes c,d,..., símbolos de función f, g,..., cada uno con una aridad asociada n>0, y símbolos de predicado R, S,..., con su aridad n≥1. Supondremos también un conjunto numerable V de variables x, y,...

1.-Definición. *Σ-términos y Σ-fórmulas.*

Dada una signatura Σ, el conjunto T_Σ de los Σ-términos, cuyos elementos típicos representaremos por t, t_1, t_2,..., y el conjunto F_Σ de las Σ-fórmulas φ, ψ, χ,..., se definen, respectivamente, por las siguientes reglas en forma de Backus-Naur:

$t ::= c$ (constante de Σ) |
x (variable de V) |
$f(t_1,\ldots,t_n)$ (f símbolo n-ario de función de Σ) |
if χ then t |
if χ then t_1 else t_2 (χ Σ-fórmula sin cuantificadores)

y:

$\varphi ::= t_1 \doteq t_2$ |
$R(t_1,\ldots,t_n)$ (R símbolo n-ario de predicado de Σ) |
Δt | $\neg\varphi$ | $\varphi\vee\psi$ | $\exists x\varphi$ | $\Delta\varphi$ ■

En la sintaxis de los términos admitimos dos nuevas construcciones a las que, abreviadamente, nos referiremos como if-then e if-then-else. Su sentido será patente cuando tratemos la semántica. Para las fórmulas, como dijimos, admitimos la aplicación del operador de definición tanto sobre términos como sobre fórmulas.

El resto de las conectivas usuales en primer orden se entenderán como abreviaturas, definidas del modo clásico; esto es: $\varphi\wedge\psi$ abrevia $\neg(\neg\varphi\vee\neg\psi)$, $\varphi\rightarrow\psi$ abrevia $\neg\varphi\vee\psi$ y $\varphi\leftrightarrow\psi$ abrevia $(\varphi\rightarrow\psi)\wedge(\psi\rightarrow\varphi)$. Así mismo, la cuantificación universal $\forall x\varphi$ abrevia $\neg\exists x\neg\varphi$.

A la lógica que hemos comenzado a introducir en la definición anterior nos referiremos abreviadamente mediante LFP (Lógica para Funciones Parciales).

Como se ha dicho en la presentación de este capítulo, la semántica de las funciones parciales se basa en su transformación en totales introduciendo un elemento nuevo apropiado. Para ello convertiremos los dominios de las estructuras en conjuntos con un elemento destacado; más exactamente, emplearemos órdenes parciales completos, abreviadamente cpo's, planos, para definir la semántica de una función parcial, siguiendo una idea bien conocida de Scott [Sco 70].

Dado pues un conjunto A, se dice que convertimos A en un cpo plano, que representamos por $A_\perp$, si consideramos el conjunto $A \cup \{\perp_A\}$ con el orden parcial definido sobre él:

$$x \leq y \Leftrightarrow x = \perp_A \vee x = y$$

para todo $x, y \in A_\perp$.

Como vemos un cpo plano no es más que un conjunto A con un orden que considera a sus elementos incomparables entre sí, salvo a uno, $\perp_A$, que es la base y es menor que el resto; a dicho elemento se le llama "bottom" del cpo y será el valor ficticio que tome una función cuando diverja. Obsérvese que, de acuerdo con esto, el cpo B definido en §1.A.1.1 y formado por $B(=\{\underline{t}, \underline{f}, \perp\})$ con su orden, se puede considerar como la conversión del conjunto de valores clásicos en un cpo plano. Abusando de la notación escribiremos B en lugar de $B_\perp$ y $\perp$ en lugar de $\perp_B$.

La semántica de las funciones parciales supone que éstas son estrictas. Con esto queremos decir que si una función está definida para ciertos argumentos, todos ellos a su vez estarán definidos. Más formalmente:

2.- Definición. *Funciones estrictas.*

Dados $A^i_\perp$, $i=1,\ldots,n,n+1$, cpos planos, y una función $f: A^1_\perp x \ldots x A^n_\perp \longrightarrow A^{n+1}_\perp$, diremos que f es estricta si verifica:

$$a_i = \perp_{Ai} \Rightarrow f(a_1,\ldots,a_n) = \perp_{An+1}$$

para todo $a_i \in A^i_\perp$, $i=1,\ldots,n$. ■

Evidentemente si f es una función parcial $f: A^1 x \ldots x A^n \dashrightarrow A^{n+1}$, donde $\dashrightarrow$ se refiere a la parcialidad de f, entonces existe $\hat{f}: A^1_\perp x \ldots x A^n_\perp \longrightarrow A^{n+1}_\perp$ definida por:

$$\hat{f}(a_1,\ldots,a_n) = \begin{cases} f(a_1,\ldots,a_n) & \text{si existe} \\ \perp_{An+1} & \text{en otro caso} \end{cases}$$

para todo $a_i \in A_\perp^i$, i=1,..,n, y $\hat{f}$ es una función estricta.

Recíprocamente, dada $f: A_\perp^1 x..xA_\perp^n \longrightarrow A_\perp^{n+1}$, estricta, existe una función parcial $f: A^1 x..xA^n - \rightarrow A^{n+1}$ definida por:

$$(a_1,\ldots,a_n) \in dom(f) \Leftrightarrow f(a_1,\ldots,a_n) \neq \perp_{An+1}$$

para todo $a_i \in A^i$, i=1,..,n.

Existe pues una correspondencia biunívoca entre funciones parciales sobre un conjunto A y funciones estrictas sobre el cpo plano $A_\perp$.

Teniendo en cuenta lo dicho hasta ahora, podemos interpretar los símbolos de una signatura Σ de la siguiente manera:

3.-Definición. *Σ-estructuras.*

Una Σ-estructura $\mathfrak{A}$ es una cuádrupla formada por un conjunto no vacío A, al que llamamos dominio de la estructura, un conjunto de elementos $c^{\mathfrak{A}} \in A_\perp$, por cada constante $c \in \Sigma$, un conjunto de funciones estrictas $f^{\mathfrak{A}}: A_\perp^n \longrightarrow A_\perp$, por cada símbolo n-ario de función $f \in \Sigma$, donde $A_\perp^n$ representa el producto cartesiano de $A_\perp$, n veces, y un conjunto de funciones estrictas $R^{\mathfrak{A}}: A_\perp^n \longrightarrow B$, por cada símbolo n-ario de predicado $R \in \Sigma$. ■

La valoración de una variable entenderemos que, lógicamente, asigna siempre a ésta un valor definido.

4.-Definición. *Valoraciones y Σ-interpretaciones.*

Dada una Σ-estructura $\mathfrak{A}$, una valoración σ sobre $\mathfrak{A}$ es una aplicación del conjunto de variables en su dominio; es decir $\sigma: V \longrightarrow A$.

Una Σ-interpretación $\mathfrak{I}$ es un par $(\mathfrak{A}, \sigma)$ formado por una Σ-estructura $\mathfrak{A}$ y una valoración σ sobre $\mathfrak{A}$. ■

La semántica de términos y fórmulas será, básicamente, la de primer orden, teniendo en cuenta las consideraciones que se hicieron en el capítulo anterior sobre el comportamiento de conectivas y cuantificadores en lógicas con tres valores booleanos.

5.-Definición. *Valor semántico de un término y una fórmula.*

Dada una Σ-interpretación $\mathfrak{I}$, la semántica de términos y fórmulas viene dada por sendas aplicaciones $T_\Sigma \longrightarrow A_\perp$ y $F_\Sigma \longrightarrow B$ que asignan a cada término t, un elemento $\mathfrak{I}(t) \in A_\perp$, y a cada fórmula φ, un valor booleano $\mathfrak{I}(\varphi) \in B$, de la manera siguiente:

(1) Términos:

$$\mathfrak{I}(c) = c^{\mathfrak{A}}$$

$$\mathfrak{I}(x) = \sigma(x)$$

$$\mathfrak{I}(f(t_1,\ldots,t_n)) = f^{\mathfrak{A}}(\mathfrak{I}(t_1),\ldots,\mathfrak{I}(t_n))$$

$$\mathfrak{I}(\text{if } \chi \text{ then t}) = \begin{cases} \mathfrak{I}(t) & \text{si } \mathfrak{I}(\chi)=\underline{t} \\ \perp_A & \text{en otro caso} \end{cases}$$

$$\mathfrak{I}(\text{if } \chi \text{ then t else t'}) = \begin{cases} \mathfrak{I}(t) & \text{si } \mathfrak{I}(\chi)=\underline{t} \\ \mathfrak{I}(t') & \text{si } \mathfrak{I}(\chi)=\underline{f} \\ \perp_A & \text{en otro caso} \end{cases}$$

(2) Fórmulas:

$$\mathfrak{I}(R(t_1,\ldots,t_n)) = R^{\mathfrak{A}}(\mathfrak{I}(t_1),\ldots,\mathfrak{I}(t_n))$$

$$\mathfrak{I}(t \approx t') = \begin{cases} \underline{t} & \text{si } \perp_A \neq \mathfrak{I}(t) = \mathfrak{I}(t') \neq \perp_A \\ \underline{f} & \text{si } \perp_A \neq \mathfrak{I}(t) \neq \mathfrak{I}(t') \neq \perp_A \\ \perp & \text{en otro caso} \end{cases}$$

$$\mathfrak{I}(\Delta t) = \begin{cases} \underline{t} & \text{si } \mathfrak{I}(t) \neq \perp_A \\ \underline{f} & \text{en otro caso} \end{cases}$$

$$\mathfrak{I}(\neg\varphi) = \begin{cases} \underline{t} & \text{si } \mathfrak{I}(\varphi) = \underline{f} \\ \underline{f} & \text{si } \mathfrak{I}(\varphi) = \underline{t} \\ \perp & \text{en otro caso} \end{cases}$$

$$\mathfrak{I}(\varphi \vee \psi) = \begin{cases} \underline{t} & \text{si } \mathfrak{I}(\varphi) = \underline{t} \text{ o } \mathfrak{I}(\psi) = \underline{t} \\ \underline{f} & \text{si } \mathfrak{I}(\varphi) = \underline{f} \text{ e } \mathfrak{I}(\psi) = \underline{f} \\ \perp & \text{en otro caso} \end{cases}$$

$$\mathfrak{I}(\exists x\varphi) = \begin{cases} \underline{t} & \text{si existe } a \in A \text{ tal que } \mathfrak{I}[a/x](\varphi) = \underline{t} \\ \underline{f} & \text{si } \mathfrak{I}[a/x](\varphi) = \underline{f} \text{ para todo } a \in A \\ \perp & \text{en otro caso} \end{cases}$$

$$\mathfrak{I}(\Delta\varphi) = \begin{cases} \underline{t} & \text{si } \mathfrak{I}(\varphi) \neq \perp \\ \perp & \text{en otro caso} \end{cases}$$

donde $\mathfrak{I}[a/x]$ es la interpretación $(\mathfrak{A}, \sigma[a/x])$, siendo $\sigma[a/x]$ la valoración definida por:

$$\sigma[a/x](y) = \begin{cases} \sigma(y) & \text{si } y \neq x \\ a & \text{si } y = x \end{cases}$$

■

Varias observaciones hay que hacer sobre la definición anterior. En primer lugar, ya que las variables siempre toman valores definidos, la fórmula Δx (resp. $\neg\Delta x$) verifica $\mathfrak{I}(\Delta x)=\underline{t}$ (resp. $\underline{f}$), para toda interpretación $\mathfrak{I}$ y toda variable x. En adelante usaremos la letra T (resp. F) para referirnos a la fórmula Δx_0 (resp. $\neg\Delta x_0$), supuesto V totalmente ordenado y siendo x_0, su primer elemento. Tendremos pues, como hemos dicho, una fórmula siempre cierta (resp. siempre falsa).

Hecho esto podemos obtener términos nunca definidos en ninguna interpretación, como por ejemplo *if* F *then* x_0, que representaremos mediante ω (no confundir con el ordinal infinito ω).

En segundo lugar, la interpretación de la igualdad $\simeq$ es la que llamaremos "débil" según la cual queda indefinida si alguno de sus miembros es un valor indefinido. En oposición se encuentra la interpretación "fuerte", 2-valorada, que asigna cierto cuando ambos miembros son iguales, es decir, indefinidos o definidos con el mismo valor, y asigna falso en otro caso. Esta interpretación, que representaremos por $\equiv$, puede expresarse en función de la débil mediante:

$$t \equiv t' :\leftrightarrow (\Delta t \wedge \Delta t' \wedge t \simeq t') \vee (\neg\Delta t \wedge \neg\Delta t')$$

Evidentemente, la débil (3-valorada) no puede expresarse en función de la fuerte (2-valorada).

Ya que disponemos de un término indefinido en toda interpretación y ya que la igualdad $\simeq$ se interpreta "débilmente" se tiene que:

$$\mathfrak{I}(\omega \simeq \omega) = \bot$$

fórmula que, siguiendo la notación del capítulo anterior, representaremos por la letra Ω. Observar que la sintaxis y semántica definidas permiten, pues, disponer de un conjunto de conectivas funcionalmente completo (cf. §1.A.2.1).

2.- RESULTADOS SEMANTICOS TECNICOS.

Algunos conceptos clásicos en lógica de primer orden pueden ser definidos aquí de una forma similar. Tal es el caso de los de aparición de una variable en un término y de una variable libre en una fórmula.

1.-Definición. *Variables en términos y variables libres en fórmulas.*

Dados un término t y una fórmula φ, se definen los conjuntos de apariciones de variables en t, escrito var(t), y de apariciones libres de variables en φ, escrito lib(φ), por inducción sobre la estructura de t y φ, como sigue:

(1) var(t) se define como en primer orden si t tiene la estructura de un término de primer orden. En los demás casos:

$\quad$ var(if χ then t) = lib(χ) $\cup$ var(t)

$\quad$ var(if χ then t else t') = lib(χ) $\cup$ var(t) $\cup$ var(t')

(2) lib(φ) se define para los nuevos tipos de fórmulas como:

$\quad$ lib(Δt) = var(t)

$\quad$ lib($\Delta\varphi$) = lib(φ) ■

Nótese que aunque T, F y Ω tienen una variable libre (x_0), sus valores en cualquier interpretación $\mathfrak{I}$ son independientes de $\mathfrak{I}(x_0)$. Lo mismo puede decirse de ω.

Obsérvese que en *if* χ *then* t, puede tomarse lib(χ) o var(χ) (concepto éste definido de forma análoga a var(t)), ya que al estar χ libre de cuantificadores, cualquier variable que aparezca en χ lo hace de forma libre.

Puede demostrarse, como en primer orden, un resultado de coincidencia (cf. [EFT 84]) para términos y fórmulas.

2.-Lema. *Coincidencia.*

Dados una estructura $\mathfrak{A}$, dos valoraciones σ y σ' sobre $\mathfrak{A}$ y un término t (resp. una fórmula φ) tal que $\sigma(x) = \sigma'(x)$, para toda variable $x \in$ var(t) (resp. lib(φ)) entonces $(\mathfrak{A},\sigma)(t) = (\mathfrak{A},\sigma')(t)$ (resp. φ). ■

Análogamente puede definirse la operación de sustitución de variables por términos mediante:

3.-Definición. *Sustitución.*

Dados una variable x y un término t, se define la sustitución de x por t en un término s, escrito s[t/x], o una fórmula φ, escrito φ[t/x], por inducción sobre la estructura de s y φ como sigue:

(1) Sustitución en términos. Para las construcciones *if-then*:

$\quad$ (if χ then s)[t/x]=if χ[t/x] then s[t/x]

$\quad$ (if χ then s else s')[t/x]=if χ[t/x] then s[t/x] else s'[t/x]

(2) Sustitución para fórmulas. Para el operador Δ aplicado a términos o fórmulas:

$$(\Delta s)[t/x] = \Delta s[t/x]$$

$$(\Delta \varphi)[t/x] = \Delta \varphi[t/x]$$

En los demás casos, la definición es la clásica. ■

Como en lógica de primer orden (cf. [EFT 84]) puede demostrarse el siguiente lema de sustitución, usando inducción sobre la estructura de términos y fórmulas:

4.-Lema. *Sustitución.*

Dados una interpretación $\mathfrak{I}$ y un término t tal que $\mathfrak{I}(t) \neq \perp_A$ se tiene, para todo término s y fórmula φ:

(a) $\mathfrak{I}(s[t/x]) = \mathfrak{I}[\mathfrak{I}(t)/x](s)$.

(b) $\mathfrak{I}(\varphi[t/x]) = \mathfrak{I}[\mathfrak{I}(t)/x](\varphi)$.

Demostración. (a) En los nuevos casos tenemos, por ejemplo:

$$\mathfrak{I}((\text{if } \chi \text{ then } s)[t/x]) = \mathfrak{I}(\text{if } \chi[t/x] \text{ then } s[t/x]) =$$

$$= \begin{cases} \mathfrak{I}(s[t/x]) & \text{si } \mathfrak{I}(\chi[t/x]) = \underline{t} \\ \perp_A & \text{en otro caso} \end{cases}$$

$$= \begin{cases} \mathfrak{I}[\mathfrak{I}(t)/x](s) & \text{si } \mathfrak{I}[\mathfrak{I}(t)/x](\chi) = \underline{t} \\ \perp_A & \text{en otro caso} \end{cases} \quad \text{(por hipótesis de inducción)}$$

$$= \mathfrak{I}[\mathfrak{I}(t)/x](\text{if } \chi \text{ then } s)$$

(b) se demuestra de forma análoga. ■

Obsérvese que exigimos que el término a sustituir t tenga valor definido por el detalle técnico de que $\mathfrak{I}[\mathfrak{I}(t)/x]$ tenga sentido y no asigne un valor indefinido a una variable lo que, como hemos dicho, no está permitido.

3.- UN METODO DE TABLEAUX PARA LFP. PRELIMINARES.

El método de los tableaux creado por Beth [Bet 59] y que nosotros formulamos en una versión inspirada en Smullyan [Smu 68] es un procedimiento de refutación extremadamente elegante y natural. Su aplicación en lógica clásica se basa en el principio de que φ es consecuencia lógica de un conjunto de fórmulas Φ si y solamente si $\Phi \cup \{\neg\varphi\}$ no es satisfactible. Por

tanto, para demostrar φ a partir de Φ basta refutar el conjunto $\Phi \cup \{\neg\varphi\}$, esto es, probar su insatisfactibilidad. Para hacerlo se construye para él un tableau que no es más que un árbol con fórmulas etiquetando sus nodos. En el proceso de su construcción, se empieza con una sucesión de nodos etiquetados con fórmulas del conjunto de partida; en cada paso, una rama puede extenderse o bifurcarse, teniendo siempre presente la idea de que si el conjunto de partida fuera satisfactible entonces existiría una rama en la que todas las fórmulas que aparecieran serían satisfactibles.

Así, si $\varphi\wedge\psi$ etiqueta la hoja de una rama, puede extenderse ésta con dos nuevos nodos etiquetados con φ y ψ, pues si $\varphi\wedge\psi$ es satisfecha, también lo son φ y ψ. Análogamente si quien etiqueta es $\varphi\vee\psi$, la rama se bifurca con dos nodos etiquetados respectivamente por φ y ψ, pues si $\varphi\vee\psi$ fuera cierta entonces alguna de las nuevas ramas terminarían en una hoja cierta.

Nosotros emplearemos el método de los tableaux para estudiar la teoría de prueba de LFP, es decir, para encontrar condiciones que permitan asegurar que un cálculo para LFP es completo. Para ello será necesario adaptar el método a las características sintácticas y semánticas que posee nuestra lógica. Una vez hecho esto explicaremos la forma de operar que tienen los tableaux para tales propósitos.

En principio, extenderemos la signatura de primer orden Σ, que supondremos a lo sumo numerable, con un conjunto numerable de nuevas constantes auxiliares $C=\{c_n \mid n<\omega\}$. El propósito de tales símbolos es servir de "testigos" en la prueba de completitud, donde seguiremos una técnica similar a la de construcción de modelo a partir de constantes (cf. [CK 73]), aplicada a un conjunto de Hintikka, obteniendo un dominio formado por clases de equivalencia de términos. La nueva signatura así obtenida, $\Sigma \cup C$, será representada mediante $\overline{\Sigma}$.

Ya que los nuevos símbolos serán usados para denotar individuos definidos con ciertas propiedades, nos restringiremos, de ahora en adelante, a aquellas $\overline{\Sigma}$-interpretaciones $\mathfrak{I}$ tales que $\mathfrak{I}(c) \neq \perp_A$, para todo $c\in C$.

Como hemos dicho, un tableau es un árbol para cuya construcción se deben dar reglas que dependen de la estructura y semántica de las fórmulas. Por ello, en primer lugar, clasificaremos el conjunto de todas las $\overline{\Sigma}$-fórmulas en clases y daremos para cada una de ellas un criterio semántico que permita asociar a una de sus fórmulas, otras más simples llamadas constituyentes.

1.-Definición. *Clases BASICA, ALFA, BETA, GAMMA y DELTA.*

Una *clasificación orientada hacia los tableaux* es una partición del conjunto de las $\bar{\Sigma}$-fórmulas en cinco clases disjuntas de manera que se satisfagan los siguientes requerimientos:

(1) BASICA: Está formada por todas aquellas fórmulas que no pertenecen a ninguna de las clases restantes. No tienen constituyentes.

(2) ALFA: Está formada por el conjunto de las llamadas fórmulas conjuntivas α que deben satisfacer:

$$Sat_s\ \Phi \cup \{\alpha\} \Rightarrow Sat_s\ \Phi \cup \{\alpha_1, \ldots, \alpha_m\}$$

para ciertos constituyentes $\alpha_1, \ldots, \alpha_m$, siendo $m \geq 1$ y finito.

Esquemáticamente representaremos esto por:

$$\frac{\alpha}{\begin{array}{l}\alpha_1 \\ \vdots \\ \alpha_m\end{array}}$$

(3) BETA: Está formada por el conjunto de las llamadas fórmulas disyuntivas β que deben satisfacer:

$$Sat_s\ \Phi \cup \{\beta\} \Rightarrow Sat_s\ \Phi \cup \{\beta_1\} \text{ o } \ldots \text{ o } Sat_s\ \Phi \cup \{\beta_m\}$$

para ciertos constituyentes $\beta_1, \ldots, \beta_m$, siendo $m \geq 1$ y finito.

Su esquema ahora es:

$$\frac{\beta}{\beta_1 \mid \ldots\ldots \mid \beta_m}$$

(4) GAMMA: Está formada por el conjunto de las llamadas fórmulas universales γ que deben satisfacer:

$$Sat_s\ \Phi \cup \{\gamma\} \Rightarrow Sat_s\ \Phi \cup \{\gamma, \gamma(t)\}$$

para todo $\bar{\Sigma}$-término t. En este caso se tienen infinitos constituyentes $\gamma(t)$ para los diferentes t.

Su esquema es:

$$\frac{\gamma}{\gamma(t)} \quad (t\ \bar{\Sigma}\text{-término})$$

(5) DELTA: Está formada por el conjunto de las llamadas fórmulas existenciales δ que deben satisfacer:

$$Sat_s\ \Phi \cup \{\delta\} \Rightarrow Sat_s\ \Phi \cup \{\delta(c)\}$$

para toda constante auxiliar $c \in C$ que no aparezca ni en Φ ni en δ. Por cada $c \in C$ es posible formar un constituyente $\delta(c)$.

En este caso el esquema es:

$$\frac{\delta}{\delta(c)}$$

En todos los apartados anteriores, Φ representa un conjunto arbitrario de $\bar{\Sigma}$-fórmulas. ■

Para nuestra lógica LFP, las condiciones de (1)-(5) se satisfacen si las cinco clases se definen como sigue:

BASICA: Δx, $\neg\Delta x$, $\neg\Delta f(t_1,\ldots,t_n)$, $\neg\Delta R(t_1,\ldots,t_n)$, Δc, $\neg\Delta c$, siendo c una constante cualquiera de $\bar{\Sigma}$.

ALFA:

$$(\alpha 1)\ \frac{R(t_1,\ldots,t_n)}{\begin{array}{c}\Delta t_1\\ \vdots\\ \Delta t_n\end{array}} \qquad (\alpha 2)\ \frac{\neg R(t_1,\ldots,t_n)}{\begin{array}{c}\Delta t_1\\ \vdots\\ \Delta t_n\end{array}} \qquad (\alpha 3)\ \frac{t_1 \approx t_2}{\begin{array}{c}\Delta t_1\\ \Delta t_2\end{array}}$$

$$(\alpha 4)\ \frac{\neg t_1 \approx t_2}{\begin{array}{c}\Delta t_1\\ \Delta t_2\end{array}} \qquad (\alpha 5)\ \frac{\neg\neg\varphi}{\varphi} \qquad (\alpha 6)\ \frac{\neg(\varphi\vee\psi)}{\begin{array}{c}\neg\varphi\\ \neg\psi\end{array}}$$

$$(\alpha 7)\ \frac{\Delta f(t_1,\ldots,t_n)}{\begin{array}{c}\Delta t_1\\ \vdots\\ \Delta t_n\end{array}} \qquad (\alpha 8)\ \frac{\Delta \text{ if } \chi \text{ then } t}{\begin{array}{c}\chi\\ \Delta t\end{array}} \qquad (\alpha 9)\ \frac{\neg\Delta\exists x\varphi}{\begin{array}{c}\exists x\neg\Delta\varphi\\ \forall x(\Delta\varphi\rightarrow\neg\varphi)\end{array}}$$

$$(\alpha 10)\ \frac{\neg\Delta\neg\varphi}{\neg\Delta\varphi} \qquad (\alpha 11)\ \frac{\neg\Delta\Delta t}{F} \qquad (\alpha 12)\ \frac{\neg\Delta\Delta\varphi}{F}$$

$$(\alpha 13)\ \frac{\Delta\Delta t}{T} \qquad (\alpha 14)\ \frac{\Delta\Delta\varphi}{T}$$

BETA:

$$(\beta 1)\ \frac{\varphi\vee\psi}{\varphi \mid \psi} \qquad (\beta 2)\ \frac{\Delta \text{ if } \chi \text{ then t else t'}}{\chi\wedge\Delta t \mid \neg\chi\wedge\Delta t'}$$

$$(\beta 3)\ \frac{\neg\Delta \text{ if } \chi \text{ then } t \text{ else } t'}{\chi\wedge\neg\Delta t \mid \neg\chi\wedge\neg\Delta t' \mid \neg\Delta\chi} \qquad (\beta 4)\ \frac{\neg\Delta \text{ if } \chi \text{ then } t}{\chi\wedge\neg\Delta t \mid \neg\chi \mid \neg\Delta\chi}$$

$$(\beta 5)\ \frac{\neg\Delta(\varphi\vee\psi)}{\neg(\Delta\varphi\vee\psi) \mid \neg(\varphi\vee\Delta\psi) \mid \neg(\Delta\varphi\vee\Delta\psi)} \qquad (\beta 6)\ \frac{\neg\Delta(t_1\simeq t_2)}{\neg\Delta t_1 \mid \neg\Delta t_2}$$

$$(\beta 7)\ \frac{\Delta\varphi}{\varphi \mid \neg\varphi} \quad (\text{si } \varphi \text{ no es de la forma } \Delta\psi \text{ ni } \Delta t)$$

GAMMA:

$$(\gamma)\ \frac{\neg\exists x\varphi}{\Delta t \longrightarrow \neg\varphi[t/x]}$$

DELTA:

$$(\delta)\ \frac{\exists x\varphi}{\varphi[c/x]}$$

Es fácil comprobar que (1)-(5) de la definición anterior dan una partición de las fórmulas. Basta seguir el siguiente esquema donde a la derecha de cada tipo de fórmula aparece la condición de la clasificación anterior en la que se incluye:

	$(t_1\simeq t_2)$	$(\alpha 3)$
	$R(t_1,\ldots,t_n)$	$(\alpha 1)$
Δt:	Δc	BASICA
	Δx	BASICA
	$\Delta f(t_1,\ldots,t_n)$	$(\alpha 7)$
	Δ if χ then t	$(\alpha 8)$
	Δ if χ then t else t'	$(\beta 2)$
$\neg\varphi$:	$\neg(t_1\simeq t_2)$	$(\alpha 4)$
	$\neg R(t_1,\ldots,t_n)$	$(\alpha 2)$
	$\neg\Delta t$: $\neg\Delta c$	BASICA
	$\neg\Delta x$	BASICA
	$\neg\Delta f(t_1,\ldots,t_n)$	BASICA
	$\neg\Delta$ if χ then t	$(\beta 4)$
	$\neg\Delta$ if χ then t else t'	$(\beta 3)$

	$\neg\neg\varphi_1$		$(\alpha 5)$
	$\neg(\varphi_1 \vee \varphi_2)$		$(\alpha 6)$
	$\neg\exists x\varphi_1$		(γ)
	$\neg\Delta\varphi_1$:	$\neg\Delta(t_1 \approx t_2)$	$(\beta 6)$
		$\neg\Delta R(t_1,\ldots,t_n)$	BASICA
		$\neg\Delta\Delta t$	$(\alpha 11)$
		$\neg\Delta\neg\psi$	$(\alpha 10)$
		$\neg\Delta(\psi_1 \vee \psi_2)$	$(\beta 5)$
		$\neg\Delta\exists x\psi$	$(\alpha 9)$
		$\neg\Delta\Delta\psi$	$(\alpha 12)$
$\varphi_1 \vee \varphi_2$			$(\beta 1)$
$\exists x\varphi$			(δ)
$\Delta\varphi$:	$\Delta(t_1 \approx t_2)$		$(\beta 7)$
	$\Delta R(t_1,\ldots,t_n)$		$(\beta 7)$
	$\Delta\Delta t$		$(\alpha 13)$
	$\Delta\neg\varphi_1$		$(\beta 7)$
	$\Delta(\varphi_1 \vee \varphi_2)$		$(\beta 7)$
	$\Delta\exists x\varphi_1$		$(\beta 7)$
	$\Delta\Delta\varphi_1$		$(\alpha 14)$

2.-Lema.

La clasificación que acabamos de dar es una clasificación orientada hacia los tableaux, en el sentido de la definición anterior.

Demostración. Veamos que se satisfacen los requerimientos semánticos (2)-(5) que aparecían en la definición anterior. Para ello demostramos que dada una $\bar{\Sigma}$-interpretación $\bar{\mathfrak{I}}$ cualquiera, se tiene:

ALFA: $\bar{\mathfrak{I}} \models_s \alpha \Rightarrow \bar{\mathfrak{I}} \models_s \alpha_1 \wedge \ldots \wedge \alpha_n$

Además "$\Leftarrow$" vale también excepto en los casos $(\alpha 1)$, $(\alpha 2)$, $(\alpha 3)$, $(\alpha 4)$ y $(\alpha 7)$.

BETA: $\bar{\mathfrak{I}} \models_s \beta \Leftrightarrow \bar{\mathfrak{I}} \models_s \beta_1 \vee \ldots \vee \beta_n$

GAMMA: $\bar{\mathfrak{I}} \models_s \gamma \Rightarrow \bar{\mathfrak{I}} \models_s \gamma(t)$, para cualquier $\bar{\Sigma}$-término t

DELTA: $\bar{\mathfrak{I}} \models_s \Phi \cup \{\delta\} \Leftrightarrow \bar{\mathfrak{I}}[a/c] \models_s \Phi \cup \{\delta(c)\}$, para algún elemento a del dominio de $\mathfrak{I}$, siempre que $c \in C$ sea nueva, esto es, no aparezca ni en Φ ni en δ.

Demostremos pues ALFA por inducción sobre la estructura de la fórmula conjuntiva. Se tiene:

(i) Los casos (α1), (α2), (α3), (α4) y (α7) se basan en el hecho de que interpretamos los símbolos de función y predicado, y la igualdad, por medio de funciones estrictas; de este modo, si ésta no diverge (que es como decir que toma valor $\underline{t}$, en (α1) y (α3), $\underline{f}$, en (α2) y (α4), o distinto de $\perp_A$, en (α7)), entonces sus argumentos han de estar definidos, es decir, han de ser ciertos todos los constituyentes y, por tanto, también su conjunción.

(ii) Para (α8) supongamos que $\mathfrak{I}$ es tal que $\mathfrak{I}(\Delta$ if χ then t)=$\underline{t}$, entonces $\mathfrak{I}$(if χ then t) $\neq \perp_A$ con lo cual $\mathfrak{I}(\chi)=\underline{t}$ e $\mathfrak{I}(t)\neq\perp_A$. El recíproco es consecuencia inmediata de la semántica dada en 1.5. De forma similar para (α10), si $\mathfrak{I}(\neg\Delta\neg\varphi)=\underline{t}$ entonces $\mathfrak{I}(\Delta\neg\varphi)=\underline{f}$ luego $\mathfrak{I}(\neg\varphi)=\perp$. Por tanto $\mathfrak{I}(\varphi)=\perp$ y se tendrá $\mathfrak{I}(\neg\Delta\varphi)=\underline{t}$. Análogamente el recíproco.

(iii) Para (α9) supongamos que $\mathfrak{I}(\neg\Delta\exists x\varphi)=\underline{t}$ entonces $\mathfrak{I}(\exists x\varphi)=\perp$ luego existe $a_0\in A$ tal que $\mathfrak{I}[a_0/x](\varphi)=\perp$ y para todo $a\in A$, $\mathfrak{I}[a/x](\varphi)\neq\underline{t}$. Por tanto existe $a_0\in A$ tal que $\mathfrak{I}[a_0/x](\neg\Delta\varphi)=\underline{t}$ y para todo $a\in A$, $\mathfrak{I}[a/x](\varphi)=\underline{f}$ o $\mathfrak{I}[a/x](\varphi)=\perp$. De lo primero se deduce que $\mathfrak{I}(\exists x\neg\Delta\varphi)=\underline{t}$ y de lo segundo que $\mathfrak{I}[a/x](\Delta\varphi\rightarrow\neg\varphi)=\underline{t}$, para todo $a\in A$ (pues es inmediato que no es falsa y si estuviera indefinida entonces $\mathfrak{I}[a/x](\Delta\varphi)=\underline{t}$, $\mathfrak{I}[a/x](\varphi)=\perp$, para algún $a\in A$, y esto entra en contradicción con lo supuesto), es decir $\mathfrak{I}(\forall x(\Delta\varphi\rightarrow\neg\varphi))=\underline{t}$. Similarmente el recíproco.

(iv) Los casos (α5), (α6), (α13) y (α14) son triviales y los casos (α11) y (α12) son vacíamente ciertos.

Para BETA se tiene:

(v) Para (β3), sea $\mathfrak{I}$ tal que $\mathfrak{I}(\neg\Delta$ if χ then t else t')=$\underline{t}$ entonces o bien $\mathfrak{I}(\chi)=\underline{t}$ e $\mathfrak{I}(t)=\perp_A$, o bien $\mathfrak{I}(\chi)=\underline{f}$ e $\mathfrak{I}(t')=\perp_A$, o bien $\mathfrak{I}(\chi)=\perp$. En cada caso es fácil deducir que el correspondiente constituyente es cierto. De forma similar se demuestra para (β2) y (β4). Para los recíprocos basta recordar que si una disyunción es cierta entonces lo es alguno de sus miembros. De la certeza de un constituyente se obtiene la de la fórmula de partida, de forma similar a lo que acabamos de ver.

(vi) Para (β5) recuérdese que si una disyunción no está definida entonces, ya que no es estricta, o bien un miembro no está definido y el otro es falso, o bien los dos están indefinidos, y esto es precisamente lo que expresan los constituyentes pues, por ejemplo, $\neg(\Delta\varphi\vee\psi)$ tiene el mismo valor semántico que $\neg\Delta\varphi\wedge\neg\psi$ (que afirma que el primer miembro de la disyunción no está definido y el segundo es falso), y $\neg(\Delta\varphi\vee\Delta\psi)$, el mismo que $\neg\Delta\varphi\wedge\neg\Delta\psi$ (que expresa que los dos miembros están indefinidos). El recíproco se demuestra de forma similar.

(vii) Para (β6) obsérvese que la igualdad es el único predicado del que sabemos, "a priori", la razón para que quede indefinido y ésta no es otra que la divergencia de alguno de sus miembros. Para el recíproco recuérdese que la interpretación de la igualdad es débil y queda indefinida si alguno de sus miembros lo está.

(viii) Los casos (β1) y (β7) son inmediatos. Obsérvese que en el caso (β7), obtenemos que $\varphi \vee \neg\varphi$ es verdadera si y solo si φ está definida. Este hecho juega en nuestra lógica un papel similar a la ley del tercio excluso en lógica clásica.

Para la demostración de GAMMA tenemos:

(ix) Sea una fórmula universal cierta, esto es, supongamos que $\overline{\mathfrak{I}}(\neg\exists x\varphi)=\underline{t}$ entonces para todo $a\in A$, $\overline{\mathfrak{I}}[a/x](\varphi)=\underline{f}$. Sea t un $\overline{\Sigma}$-término cualquiera y demostremos que $\overline{\mathfrak{I}}(\Delta t \rightarrow \neg\varphi[t/x])=\underline{t}$. Si no fuera así se tendría $\overline{\mathfrak{I}}(\Delta t)=\underline{t}$ (pues el operador de definición siempre es cierto o falso, y si fuera falso, la implicación sería cierta) y a la vez $\overline{\mathfrak{I}}(\neg\varphi[t/x])=\underline{t}$. Entonces $\overline{\mathfrak{I}}(t)=a$, $a\in A$, e $\overline{\mathfrak{I}}(\varphi[t/x])=\underline{f}$ luego, aplicando el lema de sustitución, $\overline{\mathfrak{I}}[a/x](\varphi)=\underline{f}$, contra el supuesto de que $\overline{\mathfrak{I}}(\neg\exists x\varphi)=\underline{t}$.

Por último, la demostración de DELTA es como sigue:

(x) Supongamos que las fórmulas de $\Phi \cup \{\exists x\varphi\}$ fueran ciertas en $\overline{\mathfrak{I}}$ entonces existiría $a\in A$ tal que $\overline{\mathfrak{I}}[a/x](\varphi)=\underline{t}$. Sea $c\in C$ una constante auxiliar que no aparezca en $\Phi \cup \{\exists x\varphi\}$ y consideremos la interpretación $\overline{\mathfrak{I}}[a/c]$; es evidente, como en el lema de coincidencia, que $\overline{\mathfrak{I}}[a/c]$ es modelo fuerte de Φ (por no aparecer c en Φ) y de $\varphi[c/x]$ (por no aparecer en φ y usando el lema de sustitución, teniendo en cuenta que $\overline{\mathfrak{I}}[a/c](c)=a\neq\perp_A$ e $\overline{\mathfrak{I}}[a/x](\varphi)=\underline{t}$). Luego $\Phi \cup \{\varphi[c/x]\}$ tiene a $\overline{\mathfrak{I}}[a/c]$ por modelo fuerte. Similarmente el recíproco. ■

4.- TABLEAUX PARA LFP.

Como dijimos en la sección anterior, los tableaux son usados con el propósito de refutar un conjunto de fórmulas. Demostrar por refutación que φ se obtiene del conjunto de fórmulas Φ significa "llegar a contradicción" a partir de $\Phi \cup \{\neg\varphi\}$, probando que $\Phi \cup \{\neg\varphi\}$ no admite modelos fuertes. Recuérdese que esto equivale a $\Phi \vdash \varphi$, ya que nuestra consecuencia $\vdash$ es $\vdash_{sw}$ (cf. §1.B.7).

Si en el proceso de construcción de un tableau para $\Phi \cup \{\neg\varphi\}$ llegamos a que ninguna de sus ramas es fuertemente satisfactible, el conjunto de partida

tampoco lo sería ya que, gracias al lema 3.2, una rama se va extender o bifurcar de tal forma que la satisfactibilidad fuerte se conserve en la operación.

Para definir nuestro método, hemos de explicar cómo se construyen los tableaux y cuál es el criterio para decidir que una cierta rama incluye una "contradicción obvia" que garantiza la inexistencia de modelos fuertes y autoriza a cerrar el desarrollo del tableau a lo largo de ella. En lógica clásica, una "contradicción obvia" se define como la presencia de dos fórmulas opuestas, φ y $\neg\varphi$, en la rama. Nosotros tendremos que considerar otros casos.

1.-Definición. *Conjuntos coherentes. Ramas cerradas y abiertas.*

Un conjunto de $\bar{\Sigma}$-fórmulas Φ se dice coherente si satisface las siguientes condiciones:

(1) No existe ninguna fórmula φ de manera que φ, $\neg\varphi\in\Phi$ o φ, $\neg\Delta\varphi\in\Phi$ o $\neg\varphi$, $\neg\Delta\varphi\in\Phi$.

(2) $\neg\Delta x \notin \Phi$, para ninguna variable $x\in V$.

(3) $\neg\Delta c \notin \Phi$, para niguna constante auxiliar $c\in C$.

La condición (2) garantiza, en particular, que $F\notin\Phi$. Asimismo obsérvese que ningún conjunto no coherente es fuertemente satisfactible.

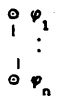
Dado un árbol Σ con fórmulas del lenguaje de LFP etiquetando sus nodos, decimos que una rama de Σ está cerrada si el conjunto de fórmulas que etiquetan sus nodos no es coherente. Una rama que no está cerrada diremos que está abierta. ■

La noción de coherencia formaliza la ausencia de "contradicciones obvias". El proceso de construcción de un tableau puede definirse ahora fácilmente, basta con ampliar las ramas que no están cerradas de acuerdo con una clasificación orientada hacia los tableaux. Ya que el proceso puede requerir realizar un número no finito de operaciones, distinguiremos entre los tableaux finitos -aquellos que obtenemos en cada momento- y las sucesiones de tableaux que definen en el límite un tableau infinito.

2.-Definición. *Tableaux finitos y sucesiones de tableaux.*

Un tableau finito para un conjunto no vacío de fórmulas Φ es un árbol definido inductivamente por las siguientes reglas:

(INI) Si $\{\varphi_1,\ldots,\varphi_n\}\subseteq\Phi$ entonces el árbol:



es un tableau finito para Φ.

(A) Si $\mathcal{I}$ es un tableau finito para Φ y α es una fórmula conjuntiva que aparece en una rama abierta de $\mathcal{I}$, entonces el árbol resultante de ampliar dicha rama con m nuevos nodos etiquetados con sus constituyentes $\alpha_1,\ldots,\alpha_m$, es un tableau finito para Φ.

(B) Si $\mathcal{I}$ es un tableau finito para Φ y β es una fórmula disyuntiva que aparece en alguna rama abierta de $\mathcal{I}$, entonces el resultado de bifurcar dicha rama con m nuevos nodos etiquetados con sus constituyentes $\beta_1,\ldots,\beta_m$, es un tableau finito para Φ.

(C) Decimos que un $\overline{\Sigma}$-término t es *adecuado* a un conjunto de fórmulas Ψ si y solo si t se puede construir con símbolos de $\overline{\Sigma}$ y variables que aparecen libres en fórmulas de Ψ, o t es la primera constante c_0 de C.

Si $\mathcal{I}$ es un tableau finito para Φ, γ es una fórmula universal que aparece en alguna rama abierta de $\mathcal{I}$ y t es un $\overline{\Sigma}$-término adecuado al conjunto de fórmulas que etiquetan dicha rama, entonces el resultado de ampliar la rama con un nuevo nodo etiquetado con el constituyente $\gamma(t)$, es un tableau finito para Φ.

(D) Si $\mathcal{I}$ es un tableau finito para Φ, δ es una fórmula existencial que aparece en alguna rama abierta de $\mathcal{I}$ y c es una constante auxiliar que no aparece en dicha rama, entonces el árbol resultante de ampliar la rama con un nuevo nodo etiquetado con $\delta(c)$, es un tableau finito para Φ.

(IG) Llamamos *axioma de la igualdad* θ a cualquier fórmula de alguno de los siguientes tipos:

$\forall x(x\simeq x)$

$\forall x\forall y(x\simeq y \rightarrow y\simeq x)$

$\forall x\forall y\forall z(x\simeq y \wedge y\simeq z \rightarrow x\simeq z)$

$\forall\bar{x}\forall\bar{y}(\bar{x}\simeq\bar{y} \wedge \Delta f(\bar{x}) \rightarrow \Delta f(\bar{y}))$

$\forall\bar{x}\forall\bar{y}(\bar{x}\simeq\bar{y} \wedge \Delta\chi(\bar{x}) \wedge \chi(\bar{x}) \rightarrow \chi(\bar{y}))$

$\forall x(\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } x)\simeq x)$

$(\chi \wedge \Delta\chi \wedge \Delta t \rightarrow (\text{if } \chi \text{ then } t \text{ else } t')\simeq t) \wedge$

$\wedge\ (\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow (\text{if } \chi \text{ then } t \text{ else } t')\simeq t')$

donde $t,\ t'\in T_{\overline{\Sigma}}$

siendo χ una fórmula libre de cuantificadores, $f \in \Sigma$, un símbolo de función n-ario y $\bar{x}$ (idem $\bar{y}$), una n-tupla de variables distintas dos a dos $x_1..x_n$, con lo que $\forall \bar{x}$ abrevia $\forall x_1..\forall x_n$ y $\bar{x} \approx \bar{y}$, $x_1 \approx y_1 \wedge .. \wedge x_n \approx y_n$. Así mismo entiéndase que con $\chi(\bar{x})$ indicamos que $lib(\chi) \subseteq \{x_1,...,x_n\}$, donde $\bar{x}$ es una n-tupla de variables, como antes, distintas dos a dos.

Un axioma de la igualdad se dice adecuado a un conjunto de fórmulas Ψ si solo usa constantes, símbolos de función y símbolos de predicado que aparezcan en Ψ, y si además todas sus variables libres aparecen libres en alguna fórmula de Ψ.

Si $\mathcal{I}$ es un tableau finito para Φ, el resultado de ampliar cualquier rama abierta con un nuevo nodo etiquetado con un axioma de la igualdad adecuado a la rama, es un tableau finito para Φ.

(HIP) Si $\mathcal{I}$ es un tableau finito para Φ, el resultado de ampliar todas las ramas abiertas de $\mathcal{I}$ con un nuevo nodo etiquetado con una fórmula de Φ que no aparezca en una de ellas, es un tableau finito para Φ.

Una sucesión de tableaux para un conjunto de fórmulas Φ es cualquier sucesión $\langle \mathcal{I}_k \ / k \in \mathbb{N} \rangle$ donde:

(i) $\mathcal{I}_0$ es algún tableau finito lineal de la forma definida en el apartado (INI) anterior.

(ii) $\mathcal{I}_{k+1}$ se obtiene a partir de $\mathcal{I}_k$ a través de alguna de las reglas que acabamos de dar en (A)-(HIP). ■

Varias observaciones hay que hacer sobre la definición anterior. En primer lugar, las siete reglas (INI)-(HIP) que acabamos de ver ponen de manifiesto cómo usar una clasificación orientada hacia los tableaux en la construcción de los mismos.

En segundo lugar, la regla (IG) corresponde al uso, en una demostración, de los axiomas de la igualdad, que aquí deben extenderse para precisar el comportamiento de los nuevos términos if-then e if-then-else y para expresar que la igualdad es compatible con la convergencia de funciones y con el carácter definido de fórmulas.

Por último hacer notar que las reglas mantienen el principio de que si $\mathcal{I}$ es un tableau finito para Φ y Φ es fuertemente satisfactible , entonces alguna rama de $\mathcal{I}$ debe ser fuertemente satisfactible. Este hecho es consecuencia de que la clasificación ALFA-DELTA satisface el lema 3.2, y será investigado detalladamente en la siguiente sección.

Como veremos en la sección 7, siempre construiremos tableaux para conjuntos no vacíos de fórmulas. Por ello, de ahora en adelante supondremos que siempre que hablemos de un tableau para Φ, éste será un conjunto no vacío de fórmulas. Queremos así evitar las molestias técnicas que supondría considerar el caso excepcional de los tableaux para el conjunto vacío $\emptyset$, para los cuales (INI) de la definición anterior no tiene sentido. Obsérvese que una definición que tuviese en cuenta $\emptyset$ podría hacerse suponiendo que todos los tableaux finitos comienzan con un nodo etiquetado por Δc_0 y se amplian como acabamos de decir.

Explicar ahora cómo se construyen tableaux infinitos no requiere más que precisar cuál es la noción de límite de una sucesión de tableaux $\langle \mathcal{T}_k \mid k \in \mathbb{N} \rangle$. Para ello "codificamos" la posición de un nodo dentro de un tableau $\mathcal{T}$ y definimos el conjunto $Pos(\mathcal{T})$ de posiciones de $\mathcal{T}$ mediante las cláusulas:

(1) 1, "código" de la raíz de $\mathcal{T}$, pertenece a $Pos(\mathcal{T})$

(2) dado $u \in \mathbb{N}$, si n es un nodo de $\mathcal{T}$ cuyo "código" es u y n' es el i-ésimo hijo de n, $i=1,\ldots,9$, la sucesión de dígitos ui es el "código" de n' y ui pertenece a $Pos(\mathcal{T})$.

Representemos entonces por $\mathcal{T}(u)$, $u \in Pos(\mathcal{T})$, la etiqueta que tiene el nodo de "código" u dentro del tableau $\mathcal{T}$. Decimos que el tableau $\mathcal{T}$ es el límite de la sucesión de tableaux $\langle \mathcal{T}_k \mid k \in \mathbb{N} \rangle$ si y solo si se verifican las siguientes condiciones:

(1) Para todo $u \in \mathbb{N}$:

$u \in Pos(\mathcal{T}) \Leftrightarrow$ existe $k \in \mathbb{N}$ tal que $u \in Pos(\mathcal{T}_k)$

(2) Para todo u, $k \in \mathbb{N}$:

$u \in Pos(\mathcal{T}_k) \Rightarrow \mathcal{T}(u) = \mathcal{T}_k(u)$.

Este árbol es el "límite" de la sucesión en el sentido del espacio métrico de los árboles (cf. [AN 80]).

3.-Definición. *Tableaux infinitos.*

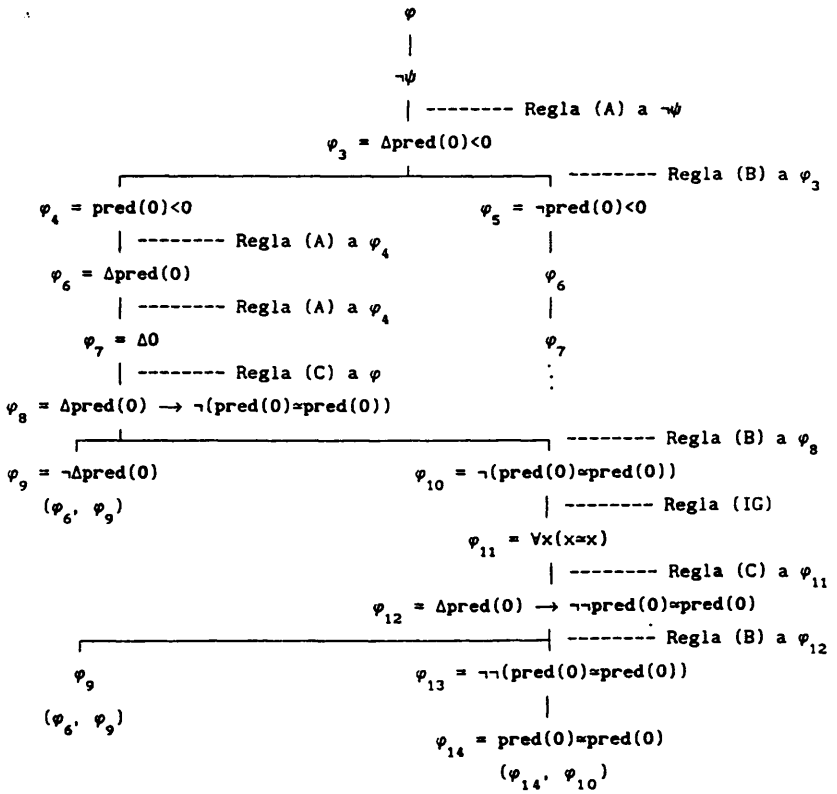
Un tableau infinito para un conjunto de fórmulas Φ se define como el límite de alguna sucesión de tableaux para Φ. ■

En lo sucesivo, al referirnos a tableaux para Φ, entenderemos que pueden ser finitos o infinitos.

Finalizaremos esta sección con un ejemplo que muestre el uso del método de los tableaux en un razonamiento dentro de la lógica para funciones parciales.

Consideremos la signatura relativa a los números naturales formada por la constante 0 y el símbolo de función unaria pred. Sea φ la fórmula que expresa que el cero no tiene predecesores, es decir: $\neg\exists x(pred(0)\approx x)$. Veamos que es posible concluir entonces que cualquier predicado que contenga el término pred(0) no está definido, en particular la fórmula ψ siguiente: $\neg\Delta pred(0)<0$.

Según sabemos, demostrar que ψ es consecuencia lógica de φ es equivalente a demostrar que $\{\varphi, \neg\psi\}$ no es fuertemente satisfactible y esto, usando el método de los tableaux, se reduce a ver que existe un tableau para $\{\varphi, \neg\psi\}$ que tiene todas sus ramas cerradas. Para ello tenemos:



El árbol empieza con el conjunto de fórmulas de partida $\{\varphi, \neg\psi\}$. Cada aplicación de una regla es explicada al margen, diciendo la fórmula a quien se aplica. La primera bifurcación conduce a dos ramas; la derecha es igual que la izquierda y, por lo tanto, no la escribiremos. La izquierda termina en tres ramas cerradas, lo que se expresa escribiendo entre paréntesis las dos fórmulas que producen la incoherencia. Como todas las ramas del tableau están cerradas, de acuerdo con lo que venimos diciendo, el conjunto de partida $\{\varphi, \neg\psi\}$ no será fuertemente satisfactible y, en consecuencia, ψ será consecuencia lógica de φ.

5.- CORRECCION DEL METODO DE LOS TABLEAUX.

La formalización del razonamiento que acabamos de hacer al final del ejemplo anterior constituye lo que venimos en llamar corrección del método. Se trata de demostrar que realmente si un conjunto de fórmulas es fuertemente satisfactible entonces no puede tener tableaux cerrados, entendiendo por tales los definidos de la manera obvia siguiente:

1.-Definición. *Tableaux abiertos y cerrados.*

Un tableau $\mathcal{T}$ se dice cerrado si es finito y todas sus ramas están cerradas. Se dice abierto si no es cerrado. ■

Podemos, entonces, expresar la corrección del método de los tableaux mediante el siguiente teorema:

2.-Teorema. *Corrección.*

Si un conjunto de fórmulas Φ tiene un tableau cerrado entonces no es fuertemente satisfactible.

Para su prueba nos bastará demostrar lo tantas veces dicho, que si el conjunto de partida es fuertemente satisfactible entonces al menos una rama de cualquier tableau finito suyo lo es. Más exactamente, se tiene:

3.-Lema.

Sea $\mathcal{T}$ un tableau finito para un conjunto de Σ-fórmulas Φ y sea $\Phi_0 \subseteq \Phi$, el subconjunto de Φ usado en la construcción de $\mathcal{T}$. Para toda Σ-interpretación $\mathcal{I}$:

$\mathcal{I}$ es modelo fuerte de Φ_0 ⇔ existen una rama en $\mathcal{T}$ con Ψ_0 como conjunto de fórmulas etiquetando sus nodos y una $\bar{\Sigma}$-interpretación $\bar{\mathcal{I}}$ expandiendo $\mathcal{I}$ tales que $\bar{\mathcal{I}}$ es modelo fuerte de Ψ_0.

Demostración. (⇒) Por inducción sobre la construcción de $\mathcal{T}$. Si se ha aplicado la regla (INI) de 4.2 entonces $\Phi_0 = \Psi_0$, tomando como rama la única que existe. El resultado es, entonces, trivial pues, por el lema de coincidencia 2.2, cualquier expansión $\bar{\mathcal{I}}$ de $\mathcal{I}$, sirve. Para el resto suponemos que existe un tableau finito $\mathcal{T}'$ al que le aplicamos una de las reglas de (A)-(HIP) para obtener $\mathcal{T}$, y que, por hipótesis de inducción, posee una rama satisfaciendo las condiciones pedidas.

Si la regla es (A), (B) o (C), el resultado se sigue de forma inmediata de las condiciones ALFA, BETA y GAMMA que se demostraron en el lema 3.2. En los tres casos vale para Σ la misma $\bar{\Sigma}$-interpretación $\bar{\mathfrak{I}}$ obtenida, por hipótesis de inducción, para Σ', pues Φ_0 es el mismo.

Para la regla (D) basta seguir un razonamiento análogo al dado en la demostración de DELTA del lema 3.2. De nuevo Φ_0 es el mismo para Σ y Σ'. Si $\bar{\mathfrak{I}}$ es la $\bar{\Sigma}$-interpretación para Σ', que nos da la hipótesis de inducción, entonces $\bar{\mathfrak{I}}[a/c]$ (cf. (x) de 3.2) vale para Σ.

El caso de la regla (HIP) es obvio pues si $\varphi\in\Phi$ es la fórmula usada para ampliar todas las ramas abiertas entonces $\varphi\in\Phi_0$ luego $\mathfrak{I}(\varphi)=\underline{t}$, luego $\bar{\mathfrak{I}}(\varphi)=\underline{t}$, aplicando el lema de coincidencia 2.2, pues $\bar{\mathfrak{I}}$ expande $\mathfrak{I}$.

Por último, para la regla (IG) basta demostrar que los axiomas de la igualdad son verdaderos en toda $\bar{\Sigma}$-interpretación $\bar{\mathfrak{I}}$. Para aquellos que expresan que la igualdad es reflexiva, simétrica y transitiva la cuestión es fácil pues, ya que las variables sólo toman valores definidos, la demostración es la misma que en lógica de primer orden.

Para el resto de los axiomas se tienen los siguientes casos. Dada $\bar{\mathfrak{I}}$ arbitraria, hemos de ver que $\bar{\mathfrak{I}} \models_s \forall\bar{x}\forall\bar{y}(\bar{x}\approx\bar{y}\wedge\Delta f(\bar{x})\rightarrow\Delta f(\bar{y}))$. Para ello es suficiente que se tenga $\bar{\mathfrak{I}}[\bar{a}/\bar{x},\ \bar{b}/\bar{y}] \models_s \bar{x}\approx\bar{y}\wedge\Delta f(\bar{x})\rightarrow\Delta f(\bar{y})$, para n-tuplas $\bar{a}$, $\bar{b}$ cualesquiera sobre el dominio de $\bar{\mathfrak{I}}$. Y en efecto, fijadas $\bar{a}$, $\bar{b}$, se tiene que: o bien $\bar{a}\neq\bar{b}$, y entonces $\bar{\mathfrak{I}}[\bar{a}/\bar{x},\ \bar{b}/\bar{y}]\ (\bar{x}\approx\bar{y}\wedge\Delta f(\bar{x})) = \underline{f}$; o bien $\bar{a}=\bar{b}$ y $f^{\bar{\mathfrak{A}}}(\bar{a})=\perp_A$, en cuyo caso también $\bar{\mathfrak{I}}[\bar{a}/\bar{x},\ \bar{b}/\bar{y}]\ (\bar{x}\approx\bar{y}\wedge\Delta f(\bar{x})) = \underline{f}$; o bien $\bar{a}=\bar{b}$ y $f^{\bar{\mathfrak{A}}}(\bar{a})\neq\perp_A$, en cuyo caso $\bar{\mathfrak{I}}[\bar{a}/\bar{x},\ \bar{b}/\bar{y}](\Delta f(\bar{y})) = \underline{t}$. En los tres casos se tiene $\bar{\mathfrak{I}}[\bar{a}/\bar{x},\ \bar{b}/\bar{y}]$ $(\bar{x}\approx\bar{y}\wedge\Delta f(\bar{x})\rightarrow\Delta f(\bar{y})) = \underline{t}$. Similarmente se demuestra, por inducción sobre la estructura de χ, que el axioma $\forall\bar{x}\forall\bar{y}(\bar{x}\approx\bar{y}\wedge\Delta\chi(\bar{x})\wedge\chi(\bar{x})\rightarrow\chi(\bar{y}))$ es cierto en cualquier $\bar{\Sigma}$-interpretación $\bar{\mathfrak{I}}$.

Respecto a los axiomas del if-then e if-then-else, el primero de ellos, $\forall x(\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } x)\approx x)$, verifica, dada $\bar{\mathfrak{I}}$ arbitraria y un elemento cualquiera a sobre el dominio de $\bar{\mathfrak{I}}$, que: o bien $\bar{\mathfrak{I}}[a/x](\Delta\chi)=\underline{f}$, con lo cual $\bar{\mathfrak{I}}[a/x](\chi\wedge\Delta\chi)=\underline{f}$; o bien $\bar{\mathfrak{I}}[a/x](\Delta\chi)=\underline{t}$ e $\bar{\mathfrak{I}}[a/x](\chi)=\underline{f}$, con lo cual $\bar{\mathfrak{I}}[a/x](\chi\wedge\Delta\chi)=\underline{f}$; o bien $\bar{\mathfrak{I}}[a/x](\Delta\chi)=\underline{t}$ e $\bar{\mathfrak{I}}[a/x](\chi)=\underline{t}$, en cuyo caso $\bar{\mathfrak{I}}[a/x](\text{if } \chi$ then $x\approx x)=\underline{t}$. En cualquiera de los casos $\bar{\mathfrak{I}}[a/x](\chi\wedge\Delta\chi\rightarrow(\text{if } \chi \text{ then } x)\approx x)=\underline{t}$. De forma similar se demuestra para $(\chi\wedge\Delta\chi\wedge\Delta t\rightarrow(\text{if } \chi \text{ then t else t'})\approx t)\wedge$

$$\wedge(\neg\chi\wedge\Delta\chi\wedge\Delta t'\rightarrow(\text{if } \chi \text{ then t else t'})\approx t')$$

considerando los dos miembros de la conjunción.

(←) Se demuestra observando que $\Phi_0 \subseteq \Psi_0$ y aplicando el lema de coincidencia (ya que $\overline{\mathfrak{I}}$ expande $\mathfrak{I}$). En efecto, sólo se introducen fórmulas de Φ en el tableau mediante la regla (HIP) de 4.2 y cuando se hace, se amplían todas las ramas abiertas. Ya que la rama que estamos considerando tiene a $\overline{\mathfrak{I}}$ como modelo fuerte entonces no puede ser cerrada, luego es abierta, luego contiene todas las fórmulas usadas cada vez que se aplicó la regla (HIP), es decir, contiene a Φ_0, como queríamos demostrar. ■

Demostración (del teorema 2). Sea $\mathcal{I}$ un tableau cerrado para Φ. Por definición de tableau cerrado, $\mathcal{I}$ es finito y solo tiene ramas cerradas. Sea Φ_0 el subconjunto finito de Φ usado en la construcción de $\mathcal{I}$. Si Φ fuese fuertemente satisfactible, existiría una Σ-interpretación $\mathfrak{I}$ que cumpliría en particular $\mathfrak{I} \models_s \Phi_0$. Por el lema 3, tendríamos también $\overline{\mathfrak{I}} \vdash \Psi_0$ para alguna $\overline{\Sigma}$-expansión $\overline{\mathfrak{I}}$ de $\mathfrak{I}$ y alguna rama de $\mathcal{I}$ que tenga Ψ_0 como conjunto de etiquetas de sus nodos. Ahora bien, como $\mathcal{I}$ es cerrado, Ψ_0 no es coherente, y es absurdo que posea un modelo fuerte. Concluimos pues que Φ tampoco es fuertemente satisfactible. ■

6.- COMPLETITUD DEL METODO DE LOS TABLEAUX.

La completitud del método de los tableaux expresa el recíproco del teorema 2 de la sección anterior, es decir, que si un conjunto de fórmulas Φ no es fuertemente satisfactible entonces es posible construir para él, un tableau cerrado. Para demostrar este resultado, daremos un procedimiento para asociar a cualquier conjunto de fórmulas Φ un tableau especial, llamado canónico, que en caso de ser abierto permite construir un modelo fuerte de Φ a partir de alguna de sus ramas. Si obtenemos tableaux canónicos con esta propiedad, la completitud estará garantizada, puesto que el tableau canónico de cualquier conjunto insatisfactible tendrá que ser cerrado.

Las ramas de los tableaux canónicos capaces de proporcionar modelos fuertes serán aquellas que, sin estar cerradas, ya no pueden prolongarse más mediante las reglas de construcción de tableaux. El conjunto de fórmulas que etiquetan los nodos de una tal rama tendrá propiedades de cierre especiales que se formalizan mediante el concepto de *conjunto de Hintikka*.

1.-Definición. *Conjunto de Hintikka.*

Un conjunto H de $\bar{\Sigma}$-fórmulas se dice de Hintikka si y solo si verifica las siguientes condiciones:

(A) Para toda fórmula $\alpha \in$ ALFA: $\alpha \in H \Rightarrow \alpha_i \in H$, para todo $i=1,\ldots,m$.

(B) Para toda fórmula $\beta \in$ BETA: $\beta \in H \Rightarrow \beta_i \in H$, para algún $i=1,\ldots,m$.

(C) Para toda fórmula $\gamma \in$ GAMMA: $\gamma \in H \Rightarrow \gamma(t) \in H$, para todo $\bar{\Sigma}$-término t adecuado al conjunto H.

(D) Para toda fórmula $\delta \in$ DELTA: $\delta \in H \Rightarrow \delta(c) \in H$, para alguna $c \in C$.

(E) Todo axioma de la igualdad adecuado a H pertenece a H.

(F) H es un conjunto coherente de fórmulas (cf. 4.1). ■

La característica esencial de los conjuntos de Hintikka es que todos son fuertemente satisfactibles. La construcción de un modelo fuerte para ellos se basa en la construcción de una $\bar{\Sigma}$-interpretación cuyo dominio es un conjunto de clases respecto a una relación de equivalencia definida sobre ciertos $\bar{\Sigma}$-términos.

En primer lugar, demostraremos un resultado técnico sobre los conjuntos de Hintikka:

2.-Lema.

Dados un conjunto de Hintikka H y un $\bar{\Sigma}$-término t adecuado a él, se tiene:

$$\neg\Delta t \notin H \Rightarrow \Delta t \in H$$

Demostración. Supongamos t en las condiciones dichas tal que $\neg\Delta t \notin H$. Usando las condiciones de la definición anterior tenemos:

$\forall x(x=x) \in H$ (por condición (E)) $\Rightarrow \neg\exists x \neg(x=x) \in H$ (abreviatura de $\forall$)

$\Rightarrow \Delta t \longrightarrow \neg\neg(t=t) \in H$ (condición (C))

$\Rightarrow \neg\Delta t \in H$ o $\neg\neg(t=t) \in H$ (condición (B))

$\Rightarrow \neg\neg(t=t) \in H$ (hipótesis)

$\Rightarrow t=t \in H$ (condición (A))

$\Rightarrow \Delta t \in H$ (condición (A)). ■

La construcción de un modelo para un conjunto de Hintikka es la habitual "al estilo de Hintikka" (cf. por ejemplo [Hin 55]) usando clases de términos para formar el dominio, con la salvedad de que ahora tomaremos clases de términos definidos.

3.-Teorema. *Existencia de modelo fuerte para conjuntos de Hintikka.*

Todo conjunto de Hintikka es fuertemente satisfactible.

Demostración. Sea $H \subseteq F_{\overline{\Sigma}}$ un conjunto de Hintikka. Consideremos el conjunto de $\overline{\Sigma}$-términos:

$$T_H = \{t \in T_{\overline{\Sigma}} \ / \ \Delta t \in H\}$$

El conjunto T_H no es vacío pues $c_0 \in C$ es adecuado a H (por 4.2.(C)) y, como H es coherente, $\neg\Delta c_0 \notin H$ luego, por el lema anterior, $\Delta c_0 \in H$.

Consideremos la relación binaria $\approx$ definida sobre $T_H \times T_H$ por:

$$t \approx s \Leftrightarrow t = s \in H$$

Usando los axiomas para la igualdad $=$, se demuestra que $\approx$ es una relación de equivalencia. Para simplificar la escritura nos referiremos a las condiciones de la definición de conjunto de Hintikka como (A),..,(F). Tenemos entonces que:

(i) Reflexividad. Dado $t \in T_H$ tenemos que $\Delta t \in H$ luego:

$\forall x(x=x) \in H$ (E) $\Rightarrow \Delta t \longrightarrow \neg\neg(t=t) \in H$ (abreviatura de $\forall$ y (C))

$\Rightarrow \neg\Delta t \in H$ o $\neg\neg(t=t) \in H$ (B)

$\Rightarrow \neg\neg(t=t) \in H$ ($\Delta t \in H$ y (F))

$\Rightarrow t=t \in H \Rightarrow t \approx t$

(ii) Simetría. Dados $t,s \in T_H$ entonces Δt, $\Delta s \in H$. Supongamos que $t \approx s$ entonces $t=s \in H$ luego:

$\forall x \forall y(x=y \longrightarrow y=x) \in H$ (E) $\Rightarrow \Delta t \longrightarrow \neg\neg\forall y(t=y \longrightarrow y=t) \in H$ (C)

$\Rightarrow \forall y(t=y \longrightarrow y=t) \in H$ ((B), $\Delta t \in H$, (F) y (A))

$\Rightarrow \Delta s \longrightarrow \neg\neg(t=s \longrightarrow s=t) \in H$ (C)

$\Rightarrow t=s \longrightarrow s=t \in H$ (de forma similar usando $\Delta s \in H$)

$\Rightarrow \neg t=s \in H$ o $s=t \in H$ (B)

$\Rightarrow s=t \in H$ ($t=s \in H$ y (F)) $\Rightarrow s \approx t$

(iii) Transitividad. Dados $t,s,u \in T_H$ entonces Δt, Δs, $\Delta u \in H$. Supongamos que $t \approx s$ y $s \approx u$ entonces $t=s$, $s=u \in H$ luego:

$\forall x \forall y \forall z(x=y \wedge y=z \longrightarrow x=z) \in H$ (E) $\Rightarrow \Delta t \longrightarrow \neg\neg\forall y \forall z(t=y \wedge y=z \longrightarrow t=z) \in H$ (C)

$\Rightarrow \forall y \forall z(t=y \wedge y=z \longrightarrow t=z) \in H$ ((B), $\Delta t \in H$, (F) y (A))

$\Rightarrow t=s \wedge s=u \longrightarrow t=u \in H$ (de forma similar a t)

$\Rightarrow \neg t=s \in H$ o $\neg s=u \in H$ o $t=u \in H$ ((A) y (B))

$\Rightarrow t=u \in H$ ($t=s$, $s=u \in H$ y (F)) $\Rightarrow t \approx u$.

Es posible demostrar también que, dados t_i, $s_i \in T_H$ tales que $t_i \approx s_i$, $i=1,..,n$, se tiene:

(iv) Para todo símbolo n-ario de función $f\in\Sigma$:

$$\Delta f(t_1,\ldots,t_n)\in H \Leftrightarrow \Delta f(s_1,\ldots,s_n)\in H$$

(v) Para toda fórmula libre de cuantificadores $\chi(x_1,\ldots,x_n)$:

$$\chi(t_1,\ldots,t_n)\in H \Leftrightarrow \chi(s_1,\ldots,s_n)\in H$$

Supongamos en (iv), sin pérdida de generalidad, que f es unaria (el caso n-ario solo complicaría la notación) entonces si $\Delta f(t)\in H$ tendremos:

$\forall x\forall y(x\approx y\wedge\Delta f(x)\rightarrow\Delta f(y))\in H$ (E) $\Leftrightarrow \Delta t\rightarrow\neg\neg\forall y(t\approx y\wedge\Delta f(t)\rightarrow\Delta f(y))\in H$ (C)

$\Leftrightarrow \forall y(t\approx y\wedge\Delta f(t)\rightarrow\Delta f(y))\in H$ ((B), (A), (F) y $\Delta t\in H$, pues $t\in T_H$)

$\Leftrightarrow t\approx s\wedge\Delta f(t)\rightarrow\Delta f(s)\in H$ (similarmente)

$\Leftrightarrow \neg t\approx s\in H$ o $\neg\Delta f(t)\in H$ o $\Delta f(s)\in H$ ((B) y (A))

$\Leftrightarrow \neg\Delta f(t)\in H$ o $\Delta f(s)\in H$ ((F) y $t\approx s\in H$, pues $t\approx s$)

Luego $\Delta f(s)\in H$, por (F), ya que $\Delta f(t)\in H$. Recíprocamente, si $\Delta f(s)\in H$, como $t\approx s$ supone $s\approx t$, por simetría, entonces, razonando de forma similar, se obtiene $\Delta f(t)\in H$.

Supongamos en (v) también, sin pérdida de generalidad, que n=1. Entonces a partir del axioma $\forall\bar{x}\forall\bar{y}(\bar{x}\approx\bar{y}\wedge\Delta\chi(\bar{x})\wedge\chi(\bar{x})\rightarrow\chi(\bar{y}))$, y como hicimos en (iv), obtenemos $\neg\Delta\chi(t)\in H$ o $\neg\chi(t)\in H$ o $\chi(s)\in H$.

Si $\chi(t)\in H$ entonces $\neg\Delta\chi(t)\in H$ o $\chi(s)\in H$, por (F). Pero también por (F) no es posible que $\neg\Delta\chi(t)\in H$ luego $\chi(s)\in H$. De forma análoga el recíproco.

Los cinco apartados que acabamos de demostrar nos permiten definir la siguiente $\bar{\Sigma}$-interpretación $\bar{\mathfrak{I}} = (\bar{\mathfrak{A}},\sigma)$ con dominio A:

$A = T_H/\approx$ (conjunto cociente). En adelante, la clase de equivalencia de t será denotada por [t]

$$c^{\bar{\mathfrak{A}}} = \begin{cases} [c] & \text{si } \Delta c\in H \\ \perp_A & \text{en otro caso} \end{cases}$$

para todo símbolo de constante $c\in\Sigma$

$c^{\bar{\mathfrak{A}}} = [c]$ para todo símbolo de constante auxiliar $c\in C$ que aparezca en H; si no, un elemento arbitrario de A.

Obsérvese que si $c\in C$ aparece en H entonces el término c es adecuado a H; como $\neg\Delta c\in H$, por (F), entonces $\Delta c\in H$, por el lema 2. Por tanto [c] tiene sentido, pues $c\in T_H$.

$$f^{\bar{\mathfrak{A}}}([t_1],\ldots,[t_n]) = \begin{cases} [f(t_1,\ldots,t_n)] & \text{si } \Delta f(t_1,\ldots,t_n)\in H \\ \perp_A & \text{en otro caso} \end{cases}$$

$f^{\bar{\mathfrak{A}}}(a_1,\ldots,a_n) = \perp_A$, si $a_i=\perp_A$, para algún $i=1,\ldots,n$

$$R^{\overline{\mathfrak{I}}}([t_1],\ldots,[t_n]) = \begin{cases} \underline{t} & \text{si } R(t_1,\ldots,t_n)\in H \\ \underline{f} & \text{si } \neg R(t_1,\ldots,t_n)\in H \\ \perp & \text{si } \neg\Delta R(t_1,\ldots,t_n)\in H \\ \text{arbitrario} & \text{en otro caso} \end{cases}$$

$R^{\overline{\mathfrak{I}}}(a_1,\ldots,a_n) = \perp$, si $a_i=\perp_A$, para algún $i=1,\ldots,n$

Por (iv) y (v), la interpretación de símbolos de función y de predicado no depende del representante elegido. Decir también que, como se ve, $f^{\overline{\mathfrak{I}}}$ y $R^{\overline{\mathfrak{I}}}$ son funciones estrictas. Además, la definición de $R^{\overline{\mathfrak{I}}}$ es correcta ya que los cuatro casos son excluyentes por la condición (F) de la definición de conjuntos de Hintikka. Finalmente digamos que $R^{\overline{\mathfrak{I}}}$ asigna un valor veritativo arbitrario cuando H no incluye suficiente información que fuerce un valor particular.

Por último la valoración σ es la siguiente:

$$\sigma(x) = \begin{cases} [x] & \text{si } \Delta x\in H \\ \text{elemento arbitrario de A} & \text{en otro caso} \end{cases}$$

Obsérvese que, como para las constantes auxiliares, $\sigma(x) = [x]$, para toda variable que aparezca libre en H (pues al aparecer libre, el término x será adecuado a H y se le podrá aplicar el lema 2).

Veamos, por fin, que $\overline{\mathfrak{I}}$ es un modelo fuerte de H para lo cual demostramos, por inducción simultánea sobre la estructura de t y φ, los siguientes tres hechos:

Hecho 3.1: Para todo $t\in T_{\overline{\Sigma}}$: $\Delta t\in H \Rightarrow \overline{\mathfrak{I}}(t)=[t]$

Hecho 3.2: Para todo $t\in T_{\overline{\Sigma}}$: $\neg\Delta t\in H \Rightarrow \overline{\mathfrak{I}}(t)=\perp_A$

Hecho 3.3: Para toda $\varphi\in T_{\overline{\Sigma}}$: $\varphi\in H \Rightarrow \overline{\mathfrak{I}}(\varphi)=\underline{t}$

La inducción requerirá una medida de complejidad que permita demostrar que una fórmula es siempre más compleja que cualquiera de sus constituyentes. Dicha medida se introduce en el apéndice 1 y allí se demuestran las propiedades que requiere. Obsérvese también allí que la complejidad de ciertas fórmulas puede ser un ordinal numerable mayor que ω y, por tanto, la inducción a la que nos estamos refiriendo es transfinita.

Demostración del Hecho 3.1.

Si t es una constante o una variable, se tiene trivialmente, por definición de $\bar{\mathfrak{I}}$.

Si t es de la forma $f(t_1,\ldots,t_n)$ entonces, si $\Delta t \in H$ tendremos que $\Delta t_1,\ldots,\Delta t_n \in H$ (por (A)) luego: $\bar{\mathfrak{I}}(t) = f^{\mathfrak{A}}([t_1],\ldots,[t_n])$ (por hipótesis de inducción; en adelante, HI) $= [f(t_1,\ldots,t_n)]$ (pues $\Delta t \in H$).

Si t es de la forma if χ then s entonces, si $\Delta t \in H$ tendremos que $\chi, \Delta s \in H$, por (A). Ahora bien:

$\forall x(\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } x) \approx x) \in H$ (E) $\Rightarrow \Delta s \rightarrow \neg\neg(\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } s) \approx s) \in H$ (C)

$\Rightarrow (\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } s) \approx s) \in H$ ((B), (A), (F) y $\Delta s \in H$)

$\Rightarrow \neg\chi \in H$ o $\neg\Delta\chi \in H$ o $(\text{if } \chi \text{ then } s) \approx s \in H$ ((B) y (A))

$\Rightarrow (\text{if } \chi \text{ then } s) \approx s \in H$ ((F) y $\chi \in H$)

$\Rightarrow [\text{if } \chi \text{ then } s] = [s]$

Por tanto $\bar{\mathfrak{I}}(t) = \bar{\mathfrak{I}}(s)$ (pues $\chi \in H$ luego $\bar{\mathfrak{I}}(\chi) = \underline{t}$, por HI) $= [s]$ (pues $\Delta s \in H$ luego $\bar{\mathfrak{I}}(s) = [s]$, por HI) $= [\text{if } \chi \text{ then } s]$.

Si t es de la forma if χ then s else s' entonces, si $\Delta t \in H$ tendremos que $\chi \wedge \Delta s \in H$ o $\neg\chi \wedge \Delta s' \in H$, por (B). Supongamos, sin pérdida de generalidad, lo primero (el otro caso es análogo). Entonces χ, $\Delta s \in H$, por (A), luego:

$(\chi \wedge \Delta\chi \wedge \Delta s \rightarrow (\text{if } \chi \text{ then } s \text{ else } s') \approx s) \wedge$

$\wedge (\neg\chi \wedge \Delta\chi \wedge \Delta s' \rightarrow (\text{if } \chi \text{ then } s \text{ else } s') \approx s') \in H$ (E) $\Rightarrow$

$\Rightarrow (\chi \wedge \Delta\chi \wedge \Delta s \rightarrow (\text{if } \chi \text{ then } s \text{ else } s') \approx s) \in H$ (A)

$\Rightarrow \neg\chi \in H$ o $\neg\Delta\chi \in H$ o $\neg\Delta s \in H$ o $(\text{if } \chi \text{ then } s \text{ else } s') \approx s \in H$ ((A) y (B))

$\Rightarrow (\text{if } \chi \text{ then } s \text{ else } s') \approx s \in H$ (F)

$\Rightarrow [\text{if } \chi \text{ then } s \text{ else } s'] = [s]$

Por tanto $\bar{\mathfrak{I}}(t) = \bar{\mathfrak{I}}(s)$ (pues $\chi \in H$ luego $\bar{\mathfrak{I}}(\chi) = \underline{t}$, por HI) $= [s]$ (pues $\Delta s \in H$ luego $\bar{\mathfrak{I}}(s) = [s]$, por HI) $= [\text{if } \chi \text{ then } s \text{ else } s']$. ■ Hecho 3.1

Demostración del Hecho 3.2.

Si t es una constante o una variable es trivial, por definición de $\bar{\mathfrak{I}}$ y (F).

Si t es de la forma $f(t_1,\ldots,t_n)$, distinguimos dos casos. Cuando $\neg\Delta t_i \in H$, para todo $i=1,\ldots,n$, entonces, por el lema 2, $\Delta t_i \in H$, para todo $i=1,\ldots,n$, luego $\bar{\mathfrak{I}}(t) = f^{\mathfrak{A}}([t_1],\ldots,[t_n])$ (pues, por HI, $\bar{\mathfrak{I}}(t_i) = [t_i]$, para todo $i=1,\ldots,n$) $= \perp_A$ (pues $\neg\Delta f(t_1,\ldots,t_n) \in H$ luego $\Delta f(t_1,\ldots,t_n) \in H$, por (F)).

Si, por el contrario, $\neg\Delta t_j \in H$, para algún $j=1,\ldots,n$, entonces $\overline{\mathfrak{I}}(t_j)=\perp_A$ (por HI), luego $\overline{\mathfrak{I}}(t) = \perp_A$, pues $f^{\overline{\mathfrak{A}}}$ es estricta.

Si t es de la forma if χ then s entonces, si $\neg\Delta t \in H$ tendremos, usando (B) y (A), que o bien $\chi \wedge \neg\Delta s \in H$, es decir χ, $\neg\Delta s \in H$, o $\neg\chi \in H$ o $\neg\Delta\chi \in H$. En el primer caso se tiene que: $\overline{\mathfrak{I}}(t) = \overline{\mathfrak{I}}(s)$ (pues $\chi \in H$ luego $\overline{\mathfrak{I}}(\chi)=\underline{t}$, por HI) $= \perp_A$ (pues $\neg\Delta s \in H$ luego $\overline{\mathfrak{I}}(s)=\perp_A$, por HI). El segundo es análogo. Para el tercero, si $\neg\Delta\chi \in H$ entonces $\overline{\mathfrak{I}}(\chi)=\perp$, por HI, con lo cual $\overline{\mathfrak{I}}(t) = \perp_A$.

Si t es de la forma if χ then s else s' basta seguir un razonamiento similar al anterior para concluir que $\overline{\mathfrak{I}}(t) = \perp_A$. ■ Hecho 3.2

Demostración del Hecho 3.3.

Se tienen los siguientes casos:

$t \approx s \in H \Rightarrow \Delta t, \Delta s \in H$ y $t \approx s$ (por (A) y definición de $\approx$)

$\Rightarrow \overline{\mathfrak{I}}(t)=[t]=[s]=\overline{\mathfrak{I}}(s)$ (HI y $t \approx s$)

$\Rightarrow \overline{\mathfrak{I}}(t \approx s)=\underline{t}$ (semántica de $\approx$)

$R(t_1,\ldots,t_n) \in H \Rightarrow \Delta t_i \in H$, para todo $i=1,\ldots,n$ (A)

$\Rightarrow \overline{\mathfrak{I}}(R(t_1,\ldots,t_n)) = R^{\overline{\mathfrak{A}}}([t_1],\ldots,[t_n])$ (HI)

$= \underline{t}$ (definición de $R^{\overline{\mathfrak{A}}}$)

$\Delta t \in H \Rightarrow \overline{\mathfrak{I}}(t) = [t]$ (HI) $\Rightarrow \overline{\mathfrak{I}}(\Delta t)=\underline{t}$ (semántica de Δ)

$\neg\varphi \in H$. Distinguimos casos según sea φ:

$\neg t \approx s \in H \Rightarrow t \approx s \in H$ y $\Delta t, \Delta s \in H$ (por (F) y (A))

$\Rightarrow \overline{\mathfrak{I}}(t)=[t]$, $\overline{\mathfrak{I}}(s)=[s]$, $[t]\neq[s]$ (HI y def. $\approx$)

$\Rightarrow \overline{\mathfrak{I}}(\neg t \approx s)=\underline{t}$ (semántica de $\approx$)

$\neg R(t_1,\ldots,t_n) \in H \Rightarrow \Delta t_i \in H$, para todo $i=1,\ldots,n$ (A)

$\Rightarrow \overline{\mathfrak{I}}(R(t_1,\ldots,t_n))=R^{\overline{\mathfrak{A}}}([t_1],\ldots,[t_n])$ (HI)

$= \underline{f}$ (definición de $R^{\overline{\mathfrak{A}}}$)

$\Rightarrow \overline{\mathfrak{I}}(\neg R(t_1,\ldots,t_n)) = \underline{t}$ (semántica de $\neg$)

$\neg\Delta t \in H \Rightarrow \overline{\mathfrak{I}}(t)=\perp_A$ (HI) $\Rightarrow \overline{\mathfrak{I}}(\neg\Delta t) = \underline{t}$ (sem. Δ)

$\neg\neg\varphi \in H \Rightarrow \varphi \in H$ (A) $\Rightarrow \overline{\mathfrak{I}}(\varphi) = \overline{\mathfrak{I}}(\neg\neg\varphi) = \underline{t}$ (HI y sem. $\neg$)

$\neg(\varphi \vee \psi) \in H \Rightarrow \neg\varphi, \neg\psi \in H$ (A) $\Rightarrow \overline{\mathfrak{I}}(\neg\varphi)=\overline{\mathfrak{I}}(\neg\psi)=\underline{t}$ (por HI)

$\Rightarrow \overline{\mathfrak{I}}(\varphi \vee \psi)=\underline{f}$ (semántica de $\vee$)

$\Rightarrow \overline{\mathfrak{I}}(\neg(\varphi \vee \psi)) = \underline{t}$ (semántica de $\neg$)

$\neg\exists x\varphi \in H \Rightarrow \Delta t \rightarrow \neg\varphi[t/x] \in H$, para todo t adecuado a H

$\Rightarrow \Delta t \rightarrow \neg\varphi[t/x] \in H$, para todo $t \in T_H$ (pues si $t \in T_H$

entonces $\Delta t \in H$ luego t es adecuado a H)

$\Rightarrow \mathfrak{I}(\Delta t)=\underline{f}$ o $\mathfrak{I}(\neg\varphi[t/x])=\underline{t}$, para todo $t \in T_H$

(semántica de Δ, $\longrightarrow$ e HI)

$\Rightarrow \mathfrak{I}(\neg\varphi[t/x])=\underline{t}$, para todo $t \in T_H$ (pues si $t \in T_H$ será $\Delta t \in H$ luego $\mathfrak{I}(\Delta t)=\underline{t}$, por HI)

$\Rightarrow \mathfrak{I}(\varphi[t/x])=\underline{f}$, para todo $t \in T_H$ (sem. de $\neg$)

$\Rightarrow \mathfrak{I}[\mathfrak{I}(t)/x](\varphi)=\underline{f}$, para todo $t \in T_H$ (por el lema de sustitución 2.4, pues $\mathfrak{I}(t)=[t]$ ya que $\Delta t \in H$ al ser $t \in T_H$)

$\Rightarrow \mathfrak{I}[a/x](\varphi)=\underline{f}$, para todo $a \in A$ (def. de A)

$\Rightarrow \mathfrak{I}(\neg\exists x\varphi) = \underline{t}$ (semántica de $\neg$ y $\exists$)

$\neg\Delta\varphi \in H$. De nuevo se tienen los siguientes casos:

$\neg\Delta(t \simeq s) \in H \Rightarrow \neg\Delta t \in H$ o $\neg\Delta s \in H$ (B)

$\Rightarrow \mathfrak{I}(t)=\perp_A$ o $\mathfrak{I}(s)=\perp_A$ (HI)

$\Rightarrow \mathfrak{I}(t \simeq s)=\perp$ (semántica de $\simeq$)

$\neg\Delta R(t_1,\ldots,t_n) \in H$. Distinguimos dos casos:

caso 1: $\neg\Delta t_i \in H$, para todo $i=1,\ldots,n$

$\Rightarrow \Delta t_i \in H$, para todo $i=1,\ldots,n$ (por el lema 2)

$\Rightarrow \mathfrak{I}(t_i)=[t_i]$, para todo $i=1,\ldots,n$ (HI)

$\Rightarrow \mathfrak{I}(R(t_1,\ldots,t_n))=R^{\mathfrak{A}}([t_1],\ldots,[t_n])=\perp$ (def. $R^{\mathfrak{A}}$)

caso 2: $\neg\Delta t_j \in H$, para algún $j=1,\ldots,n$

$\Rightarrow \mathfrak{I}(t_j)=\perp_A$, para algún $j=1,\ldots,n$ (HI)

$\Rightarrow \mathfrak{I}(R(t_1,\ldots,t_n))=\perp$ (pues $R^{\mathfrak{A}}$ es estricta)

$\neg\Delta\Delta t \in H \Rightarrow F \in H$ (A) $\Rightarrow \neg\Delta x_0 \in H$ (absurdo por (F)). Luego $\neg\Delta\Delta t \in H$ y no hay nada que demostrar

$\neg\Delta\neg\varphi \in H \Rightarrow \neg\Delta\varphi \in H$ (A) $\Rightarrow \mathfrak{I}(\varphi)=\perp$ (por HI) $\Rightarrow \mathfrak{I}(\neg\varphi)=\perp$ (semántica $\neg$) $\Rightarrow \mathfrak{I}(\neg\Delta\neg\varphi) = \underline{t}$

$\neg\Delta(\varphi\vee\psi) \in H \Rightarrow$

$\Rightarrow \neg(\Delta\varphi\vee\Delta\psi) \in H$ o $\neg(\varphi\vee\Delta\psi) \in H$ o $\neg(\psi\vee\Delta\varphi) \in H$ (B)

$\Rightarrow \neg\Delta\varphi, \neg\Delta\psi \in H$ o $\neg\varphi, \neg\Delta\psi \in H$ o $\neg\psi, \neg\Delta\varphi \in H$ (A)

$\Rightarrow$ o bien $\mathfrak{I}(\varphi)=\mathfrak{I}(\psi)=\perp$ o $\mathfrak{I}(\varphi)=\underline{f}, \mathfrak{I}(\psi)=\perp$ o $\mathfrak{I}(\psi)=\underline{f}, \mathfrak{I}(\varphi)=\perp$ (HI)

$\Rightarrow \mathfrak{I}(\varphi\vee\psi)=\perp$ (semántica de v)

$\neg\Delta\exists x\varphi \in H \Rightarrow \exists x \neg\Delta\varphi$, $\forall x(\Delta\varphi \longrightarrow \neg\varphi) \in H$ (A)

$\Rightarrow \mathfrak{I}[a_0/x](\varphi)=\perp$, para algún $a_0 \in A$ e

$\mathfrak{I}[a/x](\Delta\varphi \longrightarrow \neg\varphi)=\underline{t}$, para todo $a \in A$

(por HI y semántica de $\exists$ y $\forall$)

$\Rightarrow \overline{\mathfrak{I}}[a_0/x](\varphi)=\perp$, para algún $a_0 \in A$ y

para todo $a \in A$ tal que $\overline{\mathfrak{I}}[a/x](\varphi)\neq\perp$:

$\overline{\mathfrak{I}}[a/x](\varphi)=\underline{f}$

(por semántica de Δ, $\rightarrow$ y $\neg$)

$\Rightarrow \overline{\mathfrak{I}}(\exists x\varphi)=\perp$ (semántica de $\exists$)

$\neg\Delta\Delta\varphi \in H \Rightarrow F \in H$ (A) $\Rightarrow \neg\Delta x_0 \in H$ (absurdo por (F)).

Luego $\neg\Delta\Delta\varphi \notin H$ y no hay nada que demostrar

$\varphi \vee \psi \in H \Rightarrow \varphi \in H$ o $\psi \in H$ (B) $\Rightarrow \overline{\mathfrak{I}}(\varphi)=\underline{t}$ o $\overline{\mathfrak{I}}(\psi)=\underline{t}$ (por HI)

$\Rightarrow \overline{\mathfrak{I}}(\varphi \vee \psi) = \underline{t}$ (semántica de $\vee$)

$\exists x\varphi \in H \Rightarrow \varphi[c/x] \in H$, para alguna constante auxiliar $c \in C$ (D)

$\Rightarrow \overline{\mathfrak{I}}(\varphi[c/x])=\underline{t}$, para alguna $c \in C$ (por HI)

$\Rightarrow \overline{\mathfrak{I}}[\overline{\mathfrak{I}}(c)/x](\varphi)=\underline{t}$, para alguna $c \in C$ (por el lema de sustitución 2.4, ya que $\overline{\mathfrak{I}}(c)=[c]$)

$\Rightarrow \overline{\mathfrak{I}}[a/x](\varphi)=\underline{t}$, para algún $a \in A$ (por def. de A)

$\Rightarrow \overline{\mathfrak{I}}(\exists x\varphi) = \underline{t}$ (semántica de $\exists$)

$\Delta\varphi \in H$, φ no es de la forma Δt o $\Delta\psi$ (obsérvese que, por la semántica de Δ, $\overline{\mathfrak{I}}(\Delta\Delta t) = \overline{\mathfrak{I}}(\Delta\Delta\psi) = \underline{t}$)

$\Rightarrow \varphi \in H$ o $\neg\varphi \in H$ (B) $\Rightarrow \overline{\mathfrak{I}}(\varphi)=\underline{t}$ o $\overline{\mathfrak{I}}(\varphi)=\underline{f}$ (HI y sem. de $\neg$)

$\Rightarrow \overline{\mathfrak{I}}(\varphi)\neq\perp \Rightarrow \overline{\mathfrak{I}}(\Delta\varphi) = \underline{t}$ (semántica de Δ).

■ Hecho 3.3

El hecho 3.3 demuestra, obviamente, que la interpretación $\overline{\mathfrak{I}}$, definida más arriba, es un modelo fuerte de H. ■

Recordemos que nuestra motivación para introducir los conjuntos de Hintikka al comienzo de esta sección fue caracterizar formalmente aquellas ramas de un tableau que, pese a estar abiertas, "no pueden prolongarse más" aplicando las reglas de construcción de tableaux. Podemos ahora precisar esta idea de la siguiente forma.

4.-Definición. *Tableau completo.*

Un tableau (finito o infinito) para un conjunto de fórmulas Φ se dice completo si y solo si toda rama abierta de él es completa, esto es, tiene un conjunto H de fórmulas etiquetando sus nodos que satisface:

(1) $\Phi \subseteq H$

(2) H es un conjunto de Hintikka. ■

Obsérvese que todo tableau cerrado es trivialmente completo. Nos propusimos al principio de esta sección demostrar la llamada completitud del método de los tableaux. Según dijimos, ello lo haríamos viendo la existencia del llamado tableau canónico. Pues bien, éste no es más que un tableau completo construido siguiendo un procedimiento sistemático.

5.-Teorema. *Existencia de tableaux completos. Tableau canónico.*

Todo conjunto de fórmulas tiene un tableau completo.

Demostración. Sea $\Phi=\{\varphi_0, \varphi_1, \ldots, \varphi_n, \ldots\}$ una enumeración de un conjunto cualquiera de $\overline{\Sigma}$-fórmulas, para el cual vamos a especificar la construcción de su tableau canónico.

En primer lugar, consideremos las siguientes enumeraciones:

$V=\{x_n \mid n<\omega\}$ (variables)

$C=\{c_n \mid n<\omega\}$ (constantes auxiliares)

$\Theta=\{\theta_n \mid n<\omega\}$ (enumeración de axiomas de la igualdad de $F_{\overline{\Sigma}}$; sin pérdida de generalidad supondremos que θ_0 es $\forall x(x=x)$)

$T=\{t_n \mid n<\omega\}$ (enumeración de $T_{\overline{\Sigma}}$)

$G=\{\gamma_n \mid n<\omega\}$ (enumeración de $\overline{\Sigma}$-fórmulas universales)

Para la construcción del tableau canónico, damos un algoritmo en forma de programa while. Dicho programa es el siguiente:

```
begin
    s:=0;
    I_s := ⊙ φ_0 ;
           |
           ⊙ θ_0
    while no-completo?(I_s) do
        begin
            I'_s := abd-extensión(I_s);
            I''_s := c-extensión(I'_s);
            I_{s+1} := n-extensión(I''_s);
            s:= s+1
        end
end.
```

donde:

abd-extensión(I_s) modifica I_s ejecutando las siguientes operaciones:

(i) Extender toda rama abierta que pase por un nodo etiquetado con una fórmula conjuntiva en el nivel s de Σ_s, con m nodos etiquetados con los constituyentes de dicha fórmula.

(ii) Bifurcar toda rama abierta que pase por un nodo etiquetado con una fórmula disyuntiva en el nivel s de Σ_s, con m nodos etiquetados con los constituyentes de dicha fórmula.

(iii) Extender toda rama abierta que pase por un nodo etiquetado con una fórmula existencial δ en el nivel s de Σ_s, con un nodo etiquetado con el constituyente $\delta(c_i)$, siendo i el menor índice tal que la constante auxiliar c_i no aparece en la rama.

c-extensión(Σ'_s) modifica Σ'_s a través del siguiente proceso:

Para toda rama abierta de Σ'_s, elegir el menor par de números naturales (i, j), si lo hay, tal que γ_i aparece en la rama, t_j es adecuado al conjunto de fórmulas que etiquetan nodos de la rama y $\gamma_i(t_j)$ no aparece en la rama. Extender, entonces, la rama con un nuevo nodo etiquetado con $\gamma_i(t_j)$.

n-extensión(Σ''_s) modifica Σ''_s extendiendo todas sus ramas abiertas con nuevos nodos etiquetados con φ_s (si existe) y θ_s, si éste es adecuado a la rama.

Si el programa anterior para, Σ_s es un tableau (finito) para Φ, pues las operaciones que acabamos de mencionar se corresponden con las reglas de construcción de tableaux dadas en 4.2. Obsérvese que $\forall x(x=x)$ es un axioma adecuado a cualquier conjunto de fórmulas y por tanto Σ_0 es un tableau finito (recuérdese que Φ no es vacío). Por la misma razón, si no para, el límite de la sucesión de tableaux finitos $\langle \Sigma_s : s<\omega \rangle$ es un tableau para Φ.

Razonemos ahora que en ambos casos el tableau Σ_Φ resultante -al que denominaremos tableau canónico de Φ - es completo. En el primer caso ($\Sigma_\Phi = \Sigma_s$ para cierto s finito) ello es evidente, pues el procedimiento solo se detiene en la etapa s si Σ_s es un tableau completo. En el segundo caso, Σ_Φ es el límite de la sucesión $\langle \Sigma_s : s<\omega \rangle$, y hemos de demostrar que cualquier rama abierta de Σ_Φ -si las hay- está etiquetada por un conjunto de Hintikka H que contiene a Φ. Intuitivamente, esto está garantizado por las rutinas de abd-extensión, c-extensión y n-extensión, que disponen de infinitas etapas $s<\omega$ para incluir a lo largo de cualquier rama abierta todas las fórmulas necesarias. Más formalmente, podemos razonar por reducción al absurdo y

suponer que alguna rama abierta de Σ_Φ, etiquetada por un conjunto de fórmulas Ψ, no cumpla lo requerido. El que Ψ no contenga a Φ es imposible, porque la rutina de n-extensión se ejecuta en cada etapa. Una de las condiciones (A)-(F) de los conjuntos de Hintikka debe, pues, fallar. Estudiemos un caso; los restantes son análogos. Si falla (C), existirán índices i, j tales que $\gamma_i \in \Psi$, t_j es adecuado a Ψ y $\gamma_i(t_j) \notin \Psi$. Sea Ψ_s la parte finita de Ψ correspondiente a restringirnos a Σ_s. Para algún $s<\omega$ tendremos que $\gamma_i \in \Psi_s$ y t_j es adecuado a Ψ_s. Pero entonces $\gamma_i(t_j)$ aparecerá en $\Psi_{s'}$, para algún $s'>s$, tras ejecutarse la rutina de c-extensión durante el paso de $\Sigma_{s'-1}$ a $\Sigma_{s'}$, lo que nos lleva a contradicción. ■

Podemos ahora demostrar la completitud del método de los tableaux combinando el resultado anterior con el teorema 3.

6.-Teorema. *Completitud.*

Si un conjunto de fórmulas Φ no es fuertemente satisfactible entonces tiene un tableau cerrado.

Demostración. Veamos el recíproco y supongamos que Φ no tiene un tableau cerrado. En este caso, su tableau canónico Σ_Φ será abierto. Si Σ_Φ es finito, deberá tener alguna rama abierta, puesto que no es cerrado. Si por el contrario Σ_Φ es infinito, deberá tener alguna rama infinita, por el lema de König (cf. [Kön 26]), y esta será abierta, ya que las ramas cerradas son finitas, por construcción. En ambos casos, podemos concluir que Σ_Φ tiene al menos una rama abierta. Ahora bien, como el tableau es completo, dicha rama estará etiquetada por un conjunto de fórmulas H tal que $\Phi \subseteq H$ y H es un conjunto de Hintikka. Por el teorema 3, H es fuertemente satisfactible. Ya que $\Phi \subseteq H$ entonces Φ es, también, fuertemente satisfactible. ■

7.- EL METODO DE LOS TABLEAUX Y LOS CALCULOS DE SECUENCIAS PARA LFP.

El método de los tableaux es una herramienta potente para estudiar la teoría de prueba de una lógica. En esta sección veremos cómo hacer esto con LFP, para lo cual tenemos que encontrar la relación que existe entre los tableaux y la derivabilidad. Antes de ello precisemos de qué forma son los cálculos que estudiaremos para LFP.

A lo largo de todo este capítulo venimos estudiando propiedades de una lógica para funciones parciales, con respecto a una consecuencia: $\vdash_{sw}$, que recordemos se fijó en el capítulo anterior. Dicha consecuencia dijimos que no satisfacía la regla de corte y que, por tanto, no era transitiva. Esto significa que no es posible encontrar cálculos correctos y completos para ella, "al estilo de Hilbert". En su lugar, estudiaremos los cálculos de secuencias.

1.-Definición. *Secuencias.*

Una secuencia es un par de conjuntos finitos de fórmulas (Γ,Δ). A Γ se le llama el antecedente y a Δ, el consecuente de la secuencia. ■

En adelante utilizaremos las letras Γ, Γ',.. para representar antecedentes de secuencias. Nosotros nos ocuparemos de cálculos de secuencias en los que el consecuente solo contiene una fórmula, lo que escribiremos como (Γ,φ). Siguiendo a Ebbinghaus et al. [EFT 84], representaremos la secuencia (Γ,φ) mediante $\Gamma\ \varphi$, o simplemente φ si $\Gamma=\emptyset$, no habiendo lugar a confusión sobre quien es el consecuente, que siempre será la última fórmula. Por último, decir que usaremos comas para la unión de conjuntos de fórmulas; es decir, Γ,Γ' representará $\Gamma \cup \Gamma'$.

Para los cálculos de secuencias demostraremos la corrección y la completitud, entendidas como se precisa en la siguiente definición.

2.-Definición. *Cálculos de secuencias. Corrección y completitud.*

Un cálculo de secuencias es un conjunto de esquemas de reglas de la forma:

$$\frac{\Gamma_0\ \varphi_0,\ldots,\Gamma_n\ \varphi_n\quad (n\geq 0)}{\Gamma\ \varphi}$$

Cuando n sea 0 omitiremos la parte superior de la regla y solo escribiremos la inferior.

Una secuencia Γ φ es derivable, escrito $\Gamma\vdash\varphi$, si y solo si existe una sucesión de secuencias que termina en ella y donde cada una se ha obtenido por aplicación de alguna regla a secuencias anteriores a ella en la sucesión.

Dado un conjunto cualquiera de fórmulas $\Phi \cup \{\varphi\}$, decimos que φ es derivable a partir de Φ, y escribimos $\Phi\vdash\varphi$, si y solo si existe una parte finita $\Gamma\subseteq\Phi$ tal que $\Gamma\vdash\varphi$.

Un cálculo de secuencias se dice correcto si y solo si para todo conjunto de fórmulas $\Phi \cup \{\varphi\}$: $\Phi\vdash\varphi \Rightarrow \Phi\models\varphi$.

Un cálculo de secuencias se dice completo si y solo si para todo conjunto de fórmulas $\Phi \cup \{\varphi\}$: $\Phi\models\varphi \Rightarrow \Phi\vdash\varphi$. ■

Recordemos que la relación de consecuencia lógica que se usará a lo largo de este trabajo es $\models_{sw}$. También hacemos notar que la derivación de una secuencia Γ φ, formada por Σ-fórmulas, puede usar $\overline{\Sigma}$-fórmulas.

Veamos en primer lugar cuál es la relación que existe entre los tableaux y la derivabilidad.

3.-Lema.

Si la derivabilidad en un cálculo de secuencias $\mathfrak{C}$ satisface las siguientes ocho condiciones entonces el cálculo es completo:

(0) Para todo conjunto de Σ-fórmulas Φ y todo tableau finito $\mathcal{T}$ para Φ, si toda rama de $\mathcal{T}$ con Γ como conjunto de fórmulas etiquetando sus nodos satisface $\Gamma\vdash F$, entonces $\Pi\vdash F$, siendo Π el conjunto de fórmulas de Φ usadas en la construcción de $\mathcal{T}$.

(1) $\Gamma, \varphi, \neg\varphi \vdash F$

(2) $\Gamma, \varphi, \neg\Delta\varphi \vdash F$

(3) $\Gamma, \neg\varphi, \neg\Delta\varphi \vdash F$

(4) $\Gamma, \neg\Delta x \vdash F$, para toda variable x

(5) $\Gamma, \neg\Delta c \vdash F$, para toda constante auxiliar $c\in C$

(6) $\Gamma, \neg\varphi \vdash F \Rightarrow \Gamma \vdash \varphi$

(7) $\Gamma \vdash \varphi \Rightarrow \Gamma, \Gamma' \vdash \varphi$, para todo Γ'.

Demostración. Sea $\Phi \cup \{\varphi\}$ un conjunto cualquiera de fórmulas y supongamos que $\Phi\models\varphi$. Entonces $\Phi \cup \{\neg\varphi\}$ no es fuertemente satisfactible. Por el teorema 6.6, $\Phi \cup \{\neg\varphi\}$ tiene un tableau cerrado, que por tanto será finito igual que sus ramas. Por las condiciones (1)-(5), si Γ es el conjunto de fórmulas que etiqueta una rama cualquiera entonces $\Gamma\vdash F$ (pues todas las ramas están

cerradas). Por la condición (0), si Π es el conjunto de fórmulas de $\Phi \cup \{\neg\varphi\}$ usadas en el tableau entonces $\Pi \vdash F$. Si $\neg\varphi \in \Pi$ entonces $\Pi', \neg\varphi \vdash F$, con $\Pi' \subseteq \Phi$; en otro caso, por la condición (7): $\Pi, \neg\varphi \vdash F$, con $\Pi \subseteq \Phi$. En ambos casos, por la condición (6): $\Pi \vdash \varphi$, con $\Pi \subseteq \Phi$. Como Π es finito, pues el tableau es finito, entonces $\Phi \vdash \varphi$. ■

Como vemos, mientras que las condiciones (1)-(7) son específicas sobre la derivabilidad en un cálculo de secuencias, la condición (0) se refiere todavía a una relación general que mantienen los tableaux y la derivabilidad. En el siguiente lema tratamos de expresar dicha condición en función de los cálculos de secuencias.

4.-Lema.

Si la derivabilidad en un cálculo de secuencias $\mathfrak{C}$ satisface las siguientes condiciones:

(7) $\Gamma \vdash \varphi \Rightarrow \Gamma, \Gamma' \vdash \varphi$, para todo Γ'

(8) Para toda fórmula $\alpha \in$ ALFA: $\Gamma, \alpha_1, \ldots, \alpha_n \vdash \varphi \Rightarrow \Gamma, \alpha \vdash \varphi$

(9) Para toda fórmula $\beta \in$ BETA:

$\Gamma, \beta_1 \vdash \varphi, \ldots, \Gamma, \beta_n \vdash \varphi \Rightarrow \Gamma, \beta \vdash \varphi$

(10) Para toda fórmula $\gamma \in$ GAMMA y todo $\bar{\Sigma}$-término t:

$\Gamma, \gamma(t) \vdash \varphi \Rightarrow \Gamma, \gamma \vdash \varphi$

(11) Para toda fórmula $\delta \in$ DELTA y toda constante auxiliar c que no aparezca en $\Gamma \cup \{\delta, \varphi\}$:

$\Gamma, \delta(c) \vdash \varphi \Rightarrow \Gamma, \delta \vdash \varphi$

(12) Para todo axioma de la igualdad θ: $\Gamma, \theta \vdash \varphi \Rightarrow \Gamma \vdash \varphi$

(13) $\Gamma, \varphi \vdash \psi \Rightarrow \Gamma \vdash \varphi \rightarrow \psi$

entonces:

Para todo conjunto de Σ-fórmulas Φ, toda Σ-fórmula π y todo tableau finito Σ para Φ, si toda rama de Σ con Γ como conjunto de fórmulas etiquetando sus nodos satisface $\Gamma \vdash \pi$, entonces $\Pi \vdash \pi$, siendo Π el conjunto de fórmulas de Φ usadas en la construcción de Σ.

En particular, se satisface la condición (0) del lema anterior.

Demostración. Sea Σ un tableau finito para Φ con n ramas y sea Π el conjunto de fórmulas de Φ usadas en Σ. Sean Γ_i, $i=1,\ldots,n$, los conjuntos de fórmulas que etiquetan las n ramas de Σ. Queremos demostrar que se tiene: $\Gamma_i \vdash \pi$, $i=1,\ldots,n \Rightarrow \Pi \vdash \pi$.

La prueba es por inducción sobre la construcción de Σ, es decir, si Σ_0, Σ_1,..., Σ_k es la sucesión de tableaux finitos tales que $\Sigma=\Sigma_k$, razonaremos por inducción sobre k.

Si k=0 entonces $\Sigma_0=\Sigma$ tiene una única rama $\Gamma_1=\Pi$ y el resultado es trivial. Para k>0 hay seis casos de acuerdo con las seis reglas de construcción de tableaux finitos que se dieron en 4.2. En todos los casos definiremos n, $\Gamma_1,\ldots,\Gamma_n$, Π de Σ_k a partir de los respectivos n', Γ'_1, ..., $\Gamma'_{n'}$, Π' de Σ_{k-1}.

Supongamos pues que $\Gamma_i \vdash \pi$, i=1,..,n y veamos que $\Pi \vdash \pi$, teniendo en cuenta que esto es cierto para los respectivos de Σ_{k-1}. Se tiene entonces:

Caso 1: Σ_k se obtiene a partir de Σ_{k-1} por aplicación de la regla 4.2.(A). Supongamos, sin pérdida de generalidad, que la rama modificada es la última. Entonces:

(1.1) $\Pi = \Pi'$

(1.2) $n = n'$

(1.3) $\Gamma_n = \Gamma'_{n'} \cup \{\alpha_1,\ldots,\alpha_m\}$, para algún $\alpha\in\Gamma'_{n'}$,

(1.4) $\Gamma_i = \Gamma'_i$, i=1,..,n'-1

de manera que por (1.2), (1.3), (1.4), hipótesis y la condición (8): $\Gamma'_i \vdash \pi$, i=1,..,n'. Por hipótesis de inducción: $\Pi' \vdash \pi$, luego $\Pi \vdash \pi$, por (1.1).

Caso 2: Σ_k se obtiene a partir de Σ_{k-1} por aplicación de la regla 4.2.(B). Supongamos, de nuevo, que la rama modificada es la última. Entonces:

(2.1) $\Pi = \Pi'$

(2.2) $n = n'+m-1$

(2.3) $\Gamma_{n'+j} = \Gamma'_{n'+j} \cup \{\beta_{j+1}\}$, j=0,..,m-1, para alguna $\beta\in\Gamma'_{n'}$,

(2.4) $\Gamma_i = \Gamma'_i$, i=1,..,n'-1

de manera que por (2.2), (2.3), (2.4), hipótesis y la condición (9): $\Gamma'_i \vdash \pi$, i=1,..,n'. Por hipótesis de inducción: $\Pi' \vdash \pi$ luego $\Pi \vdash \pi$, por (2.1).

Caso 3: Σ_k se obtiene a partir de Σ_{k-1} por aplicación de la regla 4.2.(C), siendo la rama modificada, la última. Entonces:

(3.1) $\Pi = \Pi'$

(3.2) $n = n'$

(3.3) $\Gamma_n = \Gamma'_{n'} \cup \{\gamma(t)\}$, para alguna $\gamma\in\Gamma'_{n'}$, y t adecuado a $\Gamma'_{n'}$,

(3.4) $\Gamma_i = \Gamma'_i$, i=1,..,n'-1

luego por (3.2), (3.3), (3.4), hipótesis y la condición (10): $\Gamma'_i \vdash \pi$, i=1,..,n'. Por hipótesis de inducción: $\Pi' \vdash \pi$ luego $\Pi \vdash \pi$, por (3.1).

Caso 4: Σ_k se obtiene a partir de Σ_{k-1} por aplicación de la regla 4.2.(D), siendo la última, la rama modificada. Entonces:

(4.1) $\Pi = \Pi'$

(4.2) $n = n'$

(4.3) $\Gamma_n = \Gamma'_{n'} \cup \{\delta(c)\}$, para alguna $\delta \in \Gamma'_{n'}$ y alguna constante auxiliar c que no aparece en $\Gamma'_{n'}$

(4.4) $\Gamma_i = \Gamma'_i$, $i=1,..,n'-1$

luego por (4.2), (4.3), (4.4), hipótesis y la condición (11): $\Gamma'_i \vdash \pi$, $i=1,..,n'$. Por hipótesis de inducción: $\Pi' \vdash \pi$ luego $\Pi \vdash \pi$, por (4.1).

Caso 5: Σ_k se obtiene a partir de Σ_{k-1} por aplicación de la regla 4.2.(IG), siendo n' la rama modificada. Entonces:

(5.1) $\Pi = \Pi'$

(5.2) $n = n'$

(5.3) $\Gamma_n = \Gamma'_{n'} \cup \{\theta\}$, para algún axioma de la igualdad θ

(5.4) $\Gamma_i = \Gamma'_i$, $i=1,..,n'-1$

luego por (5.2), (5.3), (5.4), hipótesis y la condición (12): $\Gamma'_i \vdash \pi$, $i=1,..,n'$. Por hipótesis de inducción: $\Pi' \vdash \pi$ luego $\Pi \vdash \pi$, por (5.1).

Caso 6: Σ_k se obtiene a partir de Σ_{k-1} por aplicación de la regla 4.2.(HIP). Entonces:

(6.1) $\Pi = \Pi' \cup \{\varphi\}$

(6.2) $n = n'$

(6.3) $\Gamma_i = \Gamma'_i \cup \{\varphi\}$, si Γ'_i es abierta

(6.4) $\Gamma_i = \Gamma'_i$, si Γ'_i es cerrada

de manera que se puede razonar así:

$\Gamma_i \vdash \pi$, $i=1,..,n \Rightarrow \Gamma'_i \cup \{\varphi\} \vdash \pi$, $i=1,..,n'$ (por (6.2), (6.3), (6.4) y la condición (7)) $\Rightarrow \Gamma'_i \vdash \varphi \rightarrow \pi$, $i=1,..,n'$ (por la condición (13)) $\Rightarrow \Pi' \vdash \varphi \rightarrow \pi$ (por hipótesis de inducción, pues $\varphi \rightarrow \pi$ es una Σ-fórmula) $\Rightarrow \Pi' \cup \{\varphi\} \vdash \pi$ (por la condición (13)) $\Rightarrow \Pi \vdash \pi$ (por (6.1)). ■

Por tanto podemos demostrar la completitud de un cálculo de secuencias, sin más que aplicar los dos lemas anteriores.

5.-Teorema. *Condiciones de completitud.*

Si la derivabilidad en un cálculo de secuencias $\mathfrak{C}$ satisface las condiciones (1)-(13) de los lemas 3 y 4 entonces $\mathfrak{C}$ es completo. ■

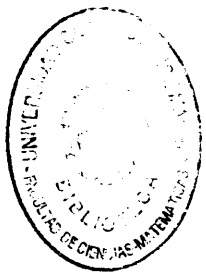
8. - UN CALCULO DE SECUENCIAS PARA LFP: $\mathfrak{GFP}$.

Las condiciones que acabamos de ver permiten obtener un cálculo de secuencias completo para LFP. Sin embargo no será muy natural, ya que estará muy influenciado por el método de los tableaux. En este sentido, la mayor parte de las pruebas serán refutaciones.

En su lugar daremos un cálculo más natural y demostraremos tanto su corrección como su completitud. Consideremos pues el cálculo de secuencias para LFP, al que llamaremos $\mathfrak{GFP}$, dado por el siguiente conjunto de reglas:

REGLAS ESTRUCTURALES

$$\text{(IHip)}\quad \frac{\Gamma \;\; \varphi}{\Gamma, \Gamma' \;\; \varphi}$$



IGUALDAD

(REF) $\quad t \approx t$

(IF) $\quad (\text{if } \chi \text{ then } t) \approx t$

(IFT) $\quad \chi \quad (\text{if } \chi \text{ then } t \text{ else } t') \approx t$

(IFE) $\quad \neg\chi \quad (\text{if } \chi \text{ then } t \text{ else } t') \approx t'$

$$\text{(Subs)}\quad \frac{\Gamma \;\; \varphi[t/x]}{\Gamma, \; t \approx t' \;\; \varphi[t'/x]}$$

$$\text{(IIg)}\quad \frac{\Gamma \;\; \neg\Delta t_i}{\Gamma \;\; t_1 \approx t_2} \qquad \frac{\Gamma \;\; \neg\Delta t_i}{\Gamma \;\; \neg(t_1 \approx t_2)} \quad (1 \le i \le 2)$$

REGLAS PARA CONECTIVAS

$$\text{(ITPr)}\quad \frac{\Gamma \;\; \neg\Delta t_i}{\Gamma \;\; R(t_1, \ldots, t_n)} \qquad \frac{\Gamma \;\; \neg\Delta t_i}{\Gamma \;\; \neg R(t_1, \ldots, t_n)} \quad (1 \le i \le n)$$

$$\text{(I}\wedge\text{C)}\quad \frac{\Gamma \;\; \neg\varphi \;,\; \Gamma \;\; \neg\psi}{\Gamma \;\; \neg(\varphi \vee \psi)} \qquad \text{(I}\vee\text{C)}\quad \frac{\Gamma \;\; \varphi}{\Gamma \;\; \varphi \vee \psi} \qquad \frac{\Gamma \;\; \varphi}{\Gamma \;\; \psi \vee \varphi}$$

$$\text{(I}\neg\neg\text{C)}\quad \frac{\Gamma \;\; \varphi}{\Gamma \;\; \neg\neg\varphi}$$

(Ctp_1) $\dfrac{\Gamma,\ \varphi\ \ \psi}{\Gamma,\ \neg\psi\ \ \neg\varphi}$ (Ctp_2) $\dfrac{\Gamma,\ \neg\varphi\ \ \neg\psi}{\Gamma,\ \psi\ \ \varphi}$

(RFT) $\dfrac{\Gamma,\ \neg\varphi\ \ F}{\Gamma\ \ \varphi}$ (MP_Δ) $\dfrac{\Gamma\ \ \varphi\rightarrow\psi,\ \ \Gamma\ \ \varphi,\ \ \Gamma\ \ \Delta\varphi}{\Gamma\ \ \psi}$

$(E\rightarrow C)$ $\dfrac{\Gamma\ \ \varphi\rightarrow\psi}{\Gamma,\ \varphi\ \ \psi}$

(Ctd) $\varphi,\ \neg\varphi\ \ \psi$

$\varphi,\ \neg\Delta\varphi\ \ \psi$

$\neg\varphi,\ \neg\Delta\varphi\ \ \psi$

$\neg\Delta x\ \ \psi$ para toda variable x

$\neg\Delta c\ \ \psi$ para toda constante auxiliar $c\in C$

REGLAS PARA CUANTIFICADORES

$(I\exists C)$ $\dfrac{\Gamma\ \ \Delta t\wedge\varphi[t/x]}{\Gamma\ \ \exists x\varphi}$ $(I\forall C)$ $\dfrac{\Gamma\ \ \neg\varphi[c/x]}{\Gamma\ \ \neg\exists x\varphi}$

donde $c\in C$ y no aparece en $\Gamma\cup\{\varphi\}$

REGLAS SOBRE DEFINICION

(IFun) $\dfrac{\Gamma\ \ \neg\Delta t_i}{\Gamma\ \ \neg\Delta f(t_1,\ldots,t_n)}$ $(1\leq i\leq n)$

(IIF) $\dfrac{\Gamma\ \ \neg\chi}{\Gamma\ \ \neg\Delta\text{if }\chi\text{ then t}}$ $\dfrac{\Gamma\ \ \neg\Delta t}{\Gamma\ \ \neg\Delta\text{if }\chi\text{ then t}}$

(DIF) $\dfrac{\Gamma\ \ \chi,\ \ \Gamma\ \ \Delta\chi,\ \ \Gamma\ \ \Delta t}{\Gamma\ \ \Delta\text{if }\chi\text{ then t}}$ (DIFT) $\dfrac{\Gamma\ \ \Delta\chi,\ \ \Gamma\ \ \chi\rightarrow\Delta t,\ \ \Gamma\ \ \neg\chi\rightarrow\Delta t'}{\Gamma\ \ \Delta\text{if }\chi\text{ then t else t'}}$

(IIFT) $\dfrac{\Gamma\ \ \chi\rightarrow\neg\Delta t,\ \ \Gamma\ \ \neg\chi\rightarrow\neg\Delta t'}{\Gamma\ \ \neg\Delta\text{if }\chi\text{ then t else t'}}$

$$(DIg)\quad \frac{\Gamma\ \Delta t,\ \Gamma\ \Delta t'}{\Gamma\ \Delta(t{=}t')} \qquad (D\neg)\quad \frac{\Gamma\ \Delta\varphi}{\Gamma\ \Delta\neg\varphi}$$

$$(Dv)\quad \frac{\Gamma\ \Delta\varphi v\psi,\ \Gamma\ \Delta\psi v\varphi,\ \Gamma\ \Delta\varphi v\Delta\psi}{\Gamma\ \Delta(\varphi v\psi)}$$

$$(D\exists)\quad \frac{\Gamma\ \neg\exists x\neg\Delta\varphi}{\Gamma\ \Delta\exists x\varphi} \qquad \frac{\Gamma\ \exists x(\Delta\varphi\wedge\varphi)}{\Gamma\ \Delta\exists x\varphi}$$

$$(IFor)\quad \frac{\Gamma\ \varphi,\ \Gamma\ \neg\varphi}{\Gamma\ \neg\Delta\varphi}$$

$$(DTer)\quad \Delta\Delta t \qquad (DFor)\quad \Delta\Delta\varphi$$

Las reglas llevan nombres nemotécnicos que indican la operación que realizan. Así, (IIFT) se refiere a la Indefinición del If-Then-else, mientras que (IIF) se refiere a la del If-then.

Obsérvese que la regla (IFor) no es la habitual de la lógica de primer orden:

$$\frac{\Gamma\ \varphi,\ \Gamma\ \neg\varphi}{\Gamma\ \psi}$$

ya que es incorrecta (basta tomar como φ la fórmula Ω y como ψ, F). Para $\vdash_{sw}$, si una fórmula φ y su negación son el consecuente de secuencias derivables, solo podemos concluir que φ no puede estar definida.

9.- REGLAS DERIVADAS Y EJEMPLOS.

Antes de demostrar la corrección y completitud del cálculo que acabamos de definir, vamos a mostrar cómo se derivan algunas reglas que emplearemos después en la prueba de completitud del cálculo. Las derivaciones serán sucesiones numeradas de secuencias con anotaciones en el margen derecho que indican las reglas que las producen y los números de las líneas donde se encuentran las hipótesis.

(Hip) $\Gamma \ \varphi$ si $\varphi \in \Gamma$

1. $\varphi, \neg\varphi \ F$ (Ctd)
2. $\Gamma, \neg\varphi \ F$ (IHip), 1, $\varphi \in \Gamma$
3. $\Gamma \ \varphi$ (RFT), 2

(E¬¬C)
$$\frac{\Gamma \ \neg\neg\varphi}{\Gamma \ \varphi}$$

1. $\Gamma \ \neg\neg\varphi$ Hipótesis
2. $\Gamma, \neg F \ \neg\neg\varphi$ (IHip), 1
3. $\Gamma, \neg\varphi \ F$ (Ctp_2), 2
4. $\Gamma \ \varphi$ (RFT), 3

(I∧C2)
$$\frac{\Gamma \ \varphi, \quad \Gamma \ \psi}{\Gamma \ \varphi\wedge\psi}$$

1. $\Gamma \ \varphi$ Hipótesis
2. $\Gamma \ \neg\neg\varphi$ (I¬¬C), 1
3. $\Gamma \ \psi$ Hipótesis
4. $\Gamma \ \neg\neg\psi$ (I¬¬C), 3
5. $\Gamma \ \neg(\neg\varphi\vee\neg\psi)$ (I∧C), 2, 4
6. $\Gamma \ \varphi\wedge\psi$ Definición de ∧, 5

(I¬∨A)
$$\frac{\Gamma, \neg\varphi, \neg\psi \ \eta}{\Gamma, \neg(\varphi\vee\psi) \ \eta}$$

1. $\Gamma, \neg\varphi, \neg\psi \ \eta$ Hipótesis
2. $\Gamma, \neg\varphi, \neg\eta \ \neg\neg\psi$ (Ctp_1), 1
3. $\Gamma, \neg\varphi, \neg\eta \ \psi$ (E¬¬C), 2
4. $\Gamma, \neg\varphi, \neg\eta \ \varphi\vee\psi$ (I∨C), 3
5. $\Gamma, \neg\varphi, \neg(\varphi\vee\psi) \ \neg\neg\eta$ (Ctp_1), 4
6. $\Gamma, \neg(\varphi\vee\psi), \neg\eta \ \varphi$ (Ctp_2), 5
7. $\Gamma, \neg(\varphi\vee\psi), \neg\eta \ \varphi\vee\psi$ (I∨C), 6
8. $\Gamma, \neg(\varphi\vee\psi) \ \neg\neg\eta$ (Ctp_1), 7
9. $\Gamma, \neg(\varphi\vee\psi) \ \eta$ (E¬¬C), 8

(I$\wedge$A) $\dfrac{\Gamma,\ \varphi,\ \psi\ \ \eta}{\Gamma,\ \varphi\wedge\psi\ \ \eta}$

1. $\Gamma,\ \varphi,\ \psi\ \ \eta$ Hipótesis
2. $\Gamma,\ \varphi,\ \neg\eta\ \ \neg\psi$ (Ctp_1), 1
3. $\Gamma,\ \varphi,\ \neg\neg\psi\ \ \neg\neg\eta$ (Ctp_1), 2
4. $\Gamma,\ \neg\neg\psi,\ \neg\neg\neg\eta\ \ \neg\varphi$ (Ctp_1), 3
5. $\Gamma,\ \neg\neg\varphi,\ \neg\neg\psi\ \ \neg\neg\neg\neg\eta$ (Ctp_1), 4
6. $\Gamma,\ \neg\neg\varphi,\ \neg\neg\psi\ \ \eta$ (E$\neg\neg$C), 5, 2 veces
7. $\Gamma,\ \neg(\neg\varphi\vee\neg\psi)\ \ \eta$ (I$\neg\vee$A), 6
8. $\Gamma,\ \varphi\wedge\psi\ \ \eta$ Definición de $\wedge$, 7

(I$\vee$A) $\dfrac{\Gamma,\ \varphi\ \ \eta,\quad \Gamma,\ \psi\ \ \eta}{\Gamma,\ \varphi\vee\psi\ \ \eta}$

1. $\Gamma,\ \varphi\ \ \eta$ Hipótesis
2. $\Gamma,\ \psi\ \ \eta$ Hipótesis
3. $\Gamma,\ \neg\eta\ \ \neg\varphi$ (Ctp_1), 1
4. $\Gamma,\ \neg\eta\ \ \neg\psi$ (Ctp_1), 2
5. $\Gamma,\ \neg\eta\ \ \neg(\varphi\vee\psi)$ (I$\wedge$C), 3, 4
6. $\Gamma,\ \varphi\vee\psi\ \ \eta$ (Ctp_2), 5

(I$\rightarrow$C) $\dfrac{\Gamma,\ \varphi\ \ \psi}{\Gamma\ \ \varphi\rightarrow\psi}$

1. $\Gamma,\ \varphi\ \ \psi$ Hipótesis
2. $\Gamma,\ \varphi,\ T\ \ \psi$ (IHip), 1
3. $\Gamma,\ \varphi,\ \neg\psi\ \ F$ (Ctp_1), 2
4. $\Gamma,\ \neg\psi,\ \neg F\ \ \neg\varphi$ (Ctp_1), 3
5. $\Gamma,\ \neg\neg\varphi,\ \neg\psi\ \ \neg\neg F$ (Ctp_1), 4
6. $\Gamma,\ \neg\neg\varphi,\ \neg\psi\ \ F$ (E$\neg\neg$C), 5
7. $\Gamma,\ \neg(\neg\varphi\vee\psi)\ \ F$ (I$\neg\vee$A), 6
8. $\Gamma\ \ \neg\varphi\vee\psi$ (RFT), 7
9. $\Gamma\ \ \varphi\rightarrow\psi$ Definición de $\rightarrow$, 8

(I$\rightarrow$C2) $\dfrac{\Gamma,\ \varphi\ \ \psi}{\Gamma\ \ \psi\vee\neg\varphi}$

1. $\Gamma,\ \varphi\ \ \psi$ Hipótesis
2. $\Gamma,\ \neg\psi,\ \neg\neg\varphi\ \ F$ Como 1-6 de (I$\rightarrow$C)
3. $\Gamma,\ \neg(\psi\vee\neg\varphi)\ \ F$ (I$\neg\vee$A), 2
4. $\Gamma\ \ \psi\vee\neg\varphi$ (RFT), 3

(DVar) $\dfrac{\Gamma,\ \Delta x\ \ \varphi}{\Gamma\ \ \varphi}$

1. $x=x$ (REF)
2. $\Gamma\ \ x=x$ (IHip), 1
3. $\neg\Delta x\ \ F$ (Ctd)
4. $\Gamma,\ \neg\Delta x\ \ F$ (IHip), 3
5. $\Gamma\ \ \Delta x$ (RFT), 4
6. $\Gamma\ \ \Delta(x=x)$ (DIg), 5
7. $\Gamma,\ \Delta x\ \ \varphi$ Hipótesis
8. $\Gamma,\ \neg\varphi\ \ \neg\Delta x$ (Ctp_1), 7
9. $\Gamma,\ \neg\varphi\ \ \neg(x=x)$ (IIg), 8
10. $\Gamma,\ x=x\ \ \varphi$ (Ctp_2), 9
11. $\Gamma\ \ x=x\rightarrow\varphi$ (I$\rightarrow$C), 10
12. $\Gamma\ \ \varphi$ (MP_Δ), 11, 2, 6

(DCon) $\dfrac{\Gamma,\ \Delta c\ \ \varphi}{\Gamma\ \ \varphi}$ donde $c\in C$

La derivación es similar a la dada para (DVar).

(IFor2) $\dfrac{\Gamma\ \ \neg\Delta\varphi}{\Gamma\ \ \varphi}$ $\dfrac{\Gamma\ \ \neg\Delta\varphi}{\Gamma\ \ \neg\varphi}$

1. $\Gamma\ \ \neg\Delta\varphi$ Hipótesis
2. $\Delta\Delta\varphi$ (DFor)
3. $\Gamma\ \ \Delta\Delta\varphi$ (IHip), 2
4. $\Gamma\ \ \Delta\neg\Delta\varphi$ (D$\neg$), 3
5. $\neg\varphi,\ \neg\Delta\varphi\ \ F$ (Ctd)

6. $\Gamma, \neg\varphi, \neg\Delta\varphi \ \ F$ (IHip), 5
7. $\Gamma, \neg\Delta\varphi \ \ \varphi$ (RFT), 6
8. $\Gamma \ \ \neg\Delta\varphi \rightarrow \varphi$ ($I\rightarrow C$), 7
9. $\Gamma \ \ \varphi$ (MP_{Δ}), 8, 1, 4

La otra regla se deriva de forma similar.

(IIFC) $\dfrac{\Gamma \ \ \neg\Delta\chi}{\Gamma \ \ \neg\Delta \text{if } \chi \text{ then } t}$

1. $\Gamma \ \ \neg\Delta\chi$ Hipótesis
2. $\Gamma \ \ \neg\chi$ (IFor2), 1
3. $\Gamma \ \ \neg\Delta \text{if } \chi \text{ then } t$ (IIF), 2

(I∃A) $\dfrac{\Gamma, \varphi[c/x] \ \ \psi}{\Gamma, \exists x\varphi \ \ \psi}$ donde $c \in C$ no aparece en $\Gamma \cup \{\varphi, \psi\}$

1. $\Gamma, \varphi[c/x] \ \ \psi$ Hipótesis
2. $\Gamma, \neg\psi \ \ \neg\varphi[c/x]$ (Ctp_1), 1
3. $\Gamma, \neg\psi \ \ \neg\exists x\varphi$ (I∀C), 2
4. $\Gamma, \exists x\varphi \ \ \psi$ (Ctp_2), 3

(I∀C2) $\dfrac{\Gamma \ \ \varphi[c/x]}{\Gamma \ \ \forall x\varphi}$ donde $c \in C$ no aparece en $\Gamma \cup \{\varphi\}$

1. $\Gamma \ \ \varphi[c/x]$ Hipótesis
2. $\Gamma \ \ \neg\neg\varphi[c/x]$ ($I\neg\neg C$), 1
3. $\Gamma \ \ \neg\exists x\neg\varphi$ (I∀C), 2
4. $\Gamma \ \ \forall x\varphi$ Definición de ∀, 3

($I\rightarrow A$) $\dfrac{\Gamma, \neg\varphi \ \ \eta, \quad \Gamma, \psi \ \ \eta}{\Gamma, \varphi\rightarrow\psi \ \ \eta}$

1. $\Gamma, \neg\varphi \ \ \eta$ Hipótesis
2. $\Gamma, \psi \ \ \eta$ Hipótesis
3. $\Gamma, \neg\varphi \vee \psi \ \ \eta$ (IvA), 1, 2
4. $\Gamma, \varphi\rightarrow\psi \ \ \eta$ Definición de $\rightarrow$, 3

(MP) $\Gamma,\ \varphi,\ \varphi\rightarrow\psi\ \ \psi$

1. $\varphi,\ \neg\varphi\ \ \psi$ (Ctd)
2. $\psi\ \ \psi$ (Hip)
3. $\varphi,\ \psi\ \ \psi$ (IHip), 2
4. $\varphi,\ \varphi\rightarrow\psi\ \ \psi$ (I$\rightarrow$A), 1, 3
5. $\Gamma,\ \varphi,\ \varphi\rightarrow\psi\ \ \psi$ (IHip), 4

(I$\neg$A) $\dfrac{\Gamma\ \ \varphi}{\Gamma,\ \neg\varphi\ \ \psi}$

1. $\Gamma\ \ \varphi$ Hipótesis
2. $\Gamma,\ \neg F,\ \neg\psi\ \ \varphi$ (IHip), 1
3. $\Gamma,\ \neg\psi,\ \neg\varphi\ \ \neg\neg F$ (Ctp_1), 2
4. $\Gamma,\ \neg\psi,\ \neg\varphi\ \ F$ (E$\neg\neg$C), 3
5. $\Gamma,\ \neg\varphi\ \ \psi$ (RFT), 4

Ley del tercio excluso (TE) $\varphi\vee\neg\varphi$

1. $\neg\varphi,\ \neg\neg\varphi\ \ F$ (Ctd)
2. $\neg(\varphi\vee\neg\varphi)\ \ F$ (I$\neg\vee$A), 1
3. $\varphi\vee\neg\varphi$ (RFT), 2

(Ctp_3) $\dfrac{\Gamma,\ \neg\varphi\ \ \psi}{\Gamma,\ \neg\psi\ \ \varphi}$

1. $\Gamma,\ \neg\varphi\ \ \psi$ Hipótesis
2. $\Gamma,\ \neg\psi\ \ \neg\neg\varphi$ (Ctp_1), 1
3. $\Gamma,\ \neg\psi\ \ \varphi$ (E$\neg\neg$C), 2

Terminamos esta sección con dos ejemplos de derivaciones llevadas a cabo en $\mathfrak{Exp}$. Empezamos formalizando el ejemplo que mostramos en la sección 4. Recordemos que allí probamos que $\varphi\vdash\psi$, siendo φ = $\neg\exists x(pred(0)\approx x)$ y ψ = $\neg\Delta(pred(0)<0)$, usando el método de los tableaux y obteniendo un tableau cerrado para $\{\varphi,\ \neg\psi\}$. Ahora podemos demostrar $\varphi\vdash\psi$ mediante la siguiente derivación:

1. $\Delta pred(0)\ \ \Delta pred(0)$ (Hip)
2. $pred(0)\approx pred(0)$ (REF)
3. $\Delta pred(0)\ \ pred(0)\approx pred(0)$ (IHip), 2

4. $\Delta pred(0)$ $\Delta pred(0) \wedge pred(0) \simeq pred(0)$ (I$\wedge$C2), 1, 3
5. $\Delta pred(0)$ $\exists x(pred(0) \simeq x)$ (I$\exists$C), 4
6. $\neg\exists x(pred(0) \simeq x)$ $\neg\Delta pred(0)$ (Ctp_1), 5
7. $\neg\exists x(pred(0) \simeq x)$ $pred(0) < 0$ (ITPr), 6
8. $\neg\exists x(pred(0) \simeq x)$ $\neg(pred(0) < 0)$ (ITPr), 6
9. $\neg\exists x(pred(0) \simeq x)$ $\neg\Delta(pred(0) < 0)$ (IFor), 7, 8

Para nuestro segundo y último ejemplo, ampliamos la signatura con un símbolo unario de función suc que hace referencia al sucesor de un número natural. Tomemos las fórmulas siguientes: $\varphi_1 = \forall x \neg(suc(x) \simeq 0)$, $\varphi_2 = \forall x(\neg x \simeq 0 \rightarrow \Delta pred(x))$ y $\psi = \forall x(\Delta pred(suc(x)))$. Demostremos que $\varphi_1, \varphi_2 \vdash \psi$ mediante la siguiente derivación; en ella, c es una constante auxiliar.

1. $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$, $\neg suc(c) \simeq 0$ $\Delta pred(suc(c))$ (MP)
2. $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$, $\neg\Delta pred(suc(c))$ $\neg\neg suc(c) \simeq 0$ (Ctp_1), 1
3. $\neg\Delta c$ $\Delta pred(suc(c))$ (Ctd)
4. $\neg\Delta c$, $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$ $\Delta pred(suc(c))$ (IHip), 3
5. $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$, $\neg\Delta pred(suc(c))$ Δc (Ctp_3), 4
6. $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$, $\neg\Delta pred(suc(c))$ $\Delta c \wedge \neg\neg suc(c) \simeq 0$ (I$\wedge$C2), 2, 5
7. $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$, $\neg\Delta pred(suc(c))$ $\exists x \neg\neg suc(x) \simeq 0$ (I$\exists$C), 6
8. $\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c))$, φ_1 $\Delta pred(suc(c))$ Def. $\forall$, (Ctp_3), 7
9. φ_1, $\neg\Delta pred(suc(c))$ $\neg(\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c)))$ (Ctp_1), 8
10. $\neg\Delta c$, $\neg\Delta suc(c)$ $\Delta pred(suc(c))$ (IHip), 3
11. $\neg\Delta suc(c)$, $\neg\Delta pred(suc(c))$ Δc (Ctp_3), 10
12. $\neg\Delta suc(c)$, $\neg\Delta pred(suc(c))$ $\neg\Delta suc(c)$ (Hip)
13. $\neg\Delta suc(c)$, $\neg\Delta pred(suc(c))$ $suc(c) \simeq 0$ (IIg), 12
14. $\neg\Delta suc(c)$, $\neg\Delta pred(suc(c))$ $\neg\neg(suc(c) \simeq 0)$ (I$\neg\neg$C), 13
15. $\neg\Delta suc(c)$, $\neg\Delta pred(suc(c))$ $\Delta c \wedge \neg\neg(suc(c) \simeq 0)$ (I$\wedge$C2), 11, 14
16. $\neg\Delta suc(c)$, $\neg\Delta pred(suc(c))$ $\exists x \neg\neg(suc(x) \simeq 0)$ (I$\exists$C), 15
17. φ_1, $\neg\Delta pred(suc(c))$ $\Delta suc(c)$ Def. $\forall$, (Ctp_3), 16
18. φ_1, $\neg\Delta pred(suc(c))$ $\Delta suc(c) \wedge \neg(\neg suc(c) \simeq 0 \rightarrow \Delta pred(suc(c)))$ (I$\wedge$C2), 9, 17
19. φ_1, $\neg\Delta pred(suc(c))$ $\exists x \neg(\neg x \simeq 0 \rightarrow \Delta pred(x))$ (I$\exists$C), 18
20. φ_1, φ_2 $\neg\neg\Delta pred(suc(c))$ Def. $\forall$, (Ctp_2), 19
21. φ_1, φ_2 ψ Def. $\forall$, (I$\forall$C), 20

10.- CORRECCION DEL CALCULO DE SECUENCIAS $\mathfrak{CJP}$.

En esta sección demostraremos la corrección del cálculo que acabamos de definir. Recordemos que la corrección de un cálculo de secuencias se establece probando que $\Phi \vdash \varphi \Rightarrow \Phi \models \varphi$, para cualquier conjunto de fórmulas dado $\Phi \cup \{\varphi\}$.

Ya que nuestro cálculo se compone de esquemas de reglas, basta demostrar que éstos son correctos, en el sentido de que preservan la consecuencia lógica, y aplicar entonces inducción sobre la longitud de la prueba formal de $\Phi \vdash \varphi$. Precisando este último concepto se tiene:

1.-Definición. *Corrección de esquemas de reglas.*

Un esquema de regla de la forma:

$$\frac{\Gamma_0 \ \varphi_0, \ldots, \Gamma_n \ \varphi_n \quad (n \geq 0)}{\Gamma \ \varphi}$$

se dice correcto si y solo si $\Gamma_i \vdash \varphi_i$ $(0 \leq i \leq n) \Rightarrow \Gamma \vdash \varphi$. ■

Tenemos entonces que:

2.-Lema.

Todas los esquemas de reglas del cálculo $\mathfrak{CJP}$ son correctos.

Demostración. Obsérvese que dada una fórmula $\alpha \in$ ALFA, si se tiene un esquema de regla de la forma: $\dfrac{\Gamma \ \neg\alpha_i}{\Gamma \ \neg\alpha}$ $(1 \leq i \leq m)$

entonces es correcto. En efecto, dada una interpretación $\overline{\mathfrak{I}}$ tal que $\overline{\mathfrak{I}}$ es modelo fuerte de Γ entonces $\overline{\mathfrak{I}}(\neg\alpha_i) \neq \underline{f}$, luego $\overline{\mathfrak{I}}(\alpha_i) \neq \underline{t}$. Por tanto $\overline{\mathfrak{I}}$ no es modelo fuerte de $\alpha_1 \wedge \ldots \wedge \alpha_m$, de manera que, por 3.2 (consultar allí la condición ALFA), $\overline{\mathfrak{I}}(\alpha) \neq \underline{t}$ luego $\overline{\mathfrak{I}}(\neg\alpha) \neq \underline{f}$, como se buscaba.

Por tanto, salvo doble negación, queda establecida la corrección de (IIg), (ITPr), (IvC), (IFun), (IIF), (D$\neg$) y (D3).

De forma similar, dada $\beta \in$ BETA, un esquema de regla de la forma:

$$\frac{\Gamma \ \neg\beta_1, \ldots, \ \Gamma \ \neg\beta_m}{\Gamma \ \neg\beta}$$

es correcto. En efecto, si $\overline{\mathfrak{I}} \models_s \Gamma$ entonces $\overline{\mathfrak{I}}(\neg\beta_i) \neq \underline{f}$, para todo $i=1,\ldots,m$. Luego $\overline{\mathfrak{I}}(\beta_i) \neq \underline{t}$, para todo $i=1,\ldots,m$. Por tanto $\overline{\mathfrak{I}}(\beta) \neq \underline{t}$, por 3.2, condición BETA, y entonces $\overline{\mathfrak{I}}(\neg\beta) \neq \underline{f}$.

Son correctas pues, eliminando o introduciendo dobles negaciones de forma conveniente, (I∧C), (DIFT), (IIFT), (DIg), (Dv) y (IFor). Respecto a (DIF), si $\mathfrak{I} \models_s \Gamma$ entonces $\mathfrak{I}(\chi)\neq\underline{f}$, $\mathfrak{I}(\Delta\chi)\neq\underline{f}$ e $\mathfrak{I}(\Delta t)\neq\underline{f}$, luego $\mathfrak{I}(\chi)=\underline{t}$ e $\mathfrak{I}(\Delta t)=\underline{t}$, con lo que $\mathfrak{I}(\Delta$if χ then t$)=\underline{t}$.

Razonando de forma similar a como lo venimos haciendo, se demuestra la corrección de (I∃C) usando la condición GAMMA de 3.2.

Respecto a (I∀C), razonemos por contraposición y supongamos que no se verifica $\Gamma \vdash \neg\exists x\varphi$, entonces existe $\mathfrak{I}$ tal que $\mathfrak{I}$ es modelo fuerte de Γ, pero tal que $\mathfrak{I}(\neg\exists x\varphi)=\underline{f}$; por tanto $\mathfrak{I}(\exists x\varphi)=\underline{t}$ y existirá $a\in A$ tal que $\mathfrak{I}[a/x](\varphi)=\underline{t}$. Tomemos $c\in C$ que no aparezca en Φ u $\{\varphi\}$ y tendremos $\mathfrak{I}[a/c](\varphi[c/x])=\underline{t}$, por el lema de sustitución 2.4. Como $\mathfrak{I} \models_s \Gamma$ entonces también $\mathfrak{I}[a/c] \models_s \Gamma$, por el lema de coincidencia 2.2, teniendo en cuenta que c no aparece en Γ. Por tanto no se verifica que $\Gamma \vdash \neg\varphi[c/x]$, como queríamos demostrar.

La corrección de los esquemas que se refieren a la igualdad, (REF), (IF), (IFT) e (IFE), es inmediata teniendo en cuenta que la igualdad se interpreta débilmente.

Algunos esquemas como (I¬¬C), (Ctd), (DTer) y (DFor), son obviamente correctos; (Subs) se demuestra correcto, usando el lema de sustitución 2.4. La corrección de (IHip), (Ctp_1) y (Ctp_2) se demuestra de forma inmediata.

Veamos, pues, la corrección del resto de las reglas. Para (RFT), supongamos Γ, $\neg\varphi \vdash F$ y sea $\mathfrak{I}$ un modelo fuerte de Γ tal que $\mathfrak{I}(\varphi)=\underline{f}$; entonces $\mathfrak{I}(\neg\varphi)=\underline{t}$ luego $\mathfrak{I}$ es modelo fuerte de $\Gamma \cup \{\neg\varphi\}$ y, por tanto, también de F, lo que es absurdo.

Para (MP_Δ), supongamos que $\Gamma \vdash \varphi\rightarrow\psi$, $\Gamma \vdash \varphi$ y $\Gamma \vdash \Delta\varphi$, y sea $\mathfrak{I}$ un modelo fuerte de Γ. Entonces $\mathfrak{I}(\Delta\varphi)\neq\underline{f}$, $\mathfrak{I}(\varphi)\neq\underline{f}$ e $\mathfrak{I}(\varphi\rightarrow\psi)\neq\underline{f}$ luego $\mathfrak{I}(\varphi)=\underline{t}$ y, por tanto, $\mathfrak{I}(\psi)\neq\underline{f}$.

Para (E→C), supongamos que $\Gamma \vdash \varphi\rightarrow\psi$ y sea $\mathfrak{I}$ un modelo fuerte de $\Gamma \cup \{\varphi\}$. Entonces $\mathfrak{I}(\varphi)=\underline{t}$ e $\mathfrak{I}(\varphi\rightarrow\psi)\neq\underline{f}$ luego $\mathfrak{I}(\psi)\neq\underline{f}$. ■

Demostrado este lema, la corrección del cálculo $\mathfrak{Cjp}$ es inmediata, como hemos dicho, por inducción sobre la longitud de la prueba.

3.-Teorema. *Corrección de* $\mathfrak{Cjp}$.

Sea $\Phi \cup \{\varphi\}$ un conjunto cualquiera de fórmulas entonces:

$$\Phi \vdash \varphi \Rightarrow \Phi \models \varphi.$$ ■

11.- COMPLETITUD DEL CALCULO DE SECUENCIAS $\mathfrak{GJP}$.

En esta sección demostramos la característica fundamental de nuestro cálculo, la completitud. Recordemos que ella se establece probando que $\Phi \vdash \varphi \Rightarrow \Phi \vdash \varphi$, para cualquier conjunto de fórmulas $\Phi \cup \{\varphi\}$. Para hacerlo nos ayudaremos del teorema 7.5 que establecía condiciones de completitud, según las cuales todo cálculo de secuencias que las satisficiese sería completo. Nosotros nos centraremos en las condiciones (8)-(13) del teorema ya que las otras son, como veremos, inmediatas.

Para establecer (8) y (9) utilizamos el siguiente resultado donde se aprecia que dichas condiciones se expresaron mediante reglas, de una forma solo ligeramente distinta.

1.-Lema.

Dado un conjunto finito de fórmulas Γ:

(a) Para toda fórmula $\alpha \in$ ALFA con constituyentes $\alpha_1, \ldots, \alpha_m$ y para todo $i=1,\ldots,m$: $\Gamma \vdash \neg\alpha_i \Rightarrow \Gamma \vdash \neg\alpha$.

(b) Para toda fórmula $\beta \in$ BETA con constituyentes $\beta_1, \ldots, \beta_m$:

$$\Gamma \vdash \neg\beta_1, \ldots, \Gamma \vdash \neg\beta_m \Rightarrow \Gamma \vdash \neg\beta$$

Demostración. (a) Basta observar que, generalmente, existe un esquema de regla que lo establece. Así, para (α1) se tiene (ITPr) y para (α2), también, usando (I$\neg\neg$C). Lo mismo cabe decir para (α3) y (α4) y el esquema (IIg).

Para las conectivas se tiene (I$\neg\neg$C) para (α5), (IFun) para (α7) e (IIF) para (α8). Usando (I$\neg\neg$C) y (E$\neg\neg$C) se prueba para (α9), con (D$\exists$), para (α6), con (IvC), y para (α10), con (D$\neg$).

Para (α11) se tiene:

1. Γ , $\neg$F	Hipótesis
2. $\Delta\Delta$t	(DTer)
3. Γ $\Delta\Delta$t	(IHip), 2
4. Γ $\neg\neg\Delta\Delta$t	(I$\neg\neg$C), 3

Nótese que esta derivación no precisa hacer uso de la hipótesis. Para (α12) el razonamiento es análogo.

Para (α13) tenemos:

1. Γ $\neg$T	Hipótesis
2. Γ F	Definición de F, 1
3. Γ, $\neg\neg\Delta\Delta$t F	(IHip), 2
4. Γ $\neg\Delta\Delta$t	(RFT), 3

y, similarmente, (α14).

(b) Lo dicho para el apartado anterior vale también para éste. Obsérvese que (I$\wedge$C) establece (β1) e (IIFT) y (IFor) establecen (β2) y (β7), respectivamente, usando (E$\neg\neg$C) para esta última.

Para (β3) tenemos:

1. Γ $\neg(\chi\wedge\neg\Delta t)$ Hipótesis
2. Γ $\neg(\neg\chi\wedge\neg\Delta t')$ Hipótesis
3. Γ $\neg\neg\Delta\chi$ Hipótesis
4. Γ $\neg\neg(\neg\chi\vee\neg\neg\Delta t)$ Definición de $\wedge$, 1
5. Γ $\neg\neg(\neg\neg\chi\vee\neg\neg\Delta t')$ Definición de $\wedge$, 2
6. Γ $\neg\chi\vee\neg\neg\Delta t$ (E$\neg\neg$C), 4
7. Γ $\neg\neg\chi\vee\neg\neg\Delta t'$ (E$\neg\neg$C), 5
8. Γ $\Delta\chi$ (E$\neg\neg$C), 3
9. Γ, χ $\neg\neg\Delta t$ (E$\rightarrow$C), Definición de $\rightarrow$, 6
10. $\Gamma, \neg\chi$ $\neg\neg\Delta t'$ (E$\rightarrow$C), Definición de $\rightarrow$, 7
11. Γ, χ Δt (E$\neg\neg$C), 9
12. $\Gamma, \neg\chi$ $\Delta t'$ (E$\neg\neg$C), 10
13. Γ $\chi\rightarrow\Delta t$ (I$\rightarrow$C), 11
14. Γ $\neg\chi\rightarrow\Delta t'$ (I$\rightarrow$C), 12
15. Γ Δif χ then t else t' (DIFT), 8, 13, 14
16. Γ $\neg\neg\Delta$if χ then t else t' (I$\neg\neg$C), 15

Para (β4) se tiene:

1. Γ $\neg(\chi\wedge\neg\Delta t)$ Hipótesis
2. Γ $\neg\neg\chi$ Hipótesis
3. Γ $\neg\neg\Delta\chi$ Hipótesis
4. Γ $\neg\neg(\neg\chi\vee\neg\neg\Delta t)$ Definición de $\wedge$, 1
5. Γ $\chi\rightarrow\Delta t$ Análogo a 6, 9, 11 y 13 anteriores
6. Γ χ (E$\neg\neg$C), 2
7. Γ $\Delta\chi$ (E$\neg\neg$C), 3
8. Γ Δt (MP_{Δ}), 5, 6, 7
9. Γ Δif χ then t (DIF), 6, 7, 8
10. Γ $\neg\neg\Delta$if χ then t (I$\neg\neg$C), 9

Por último, para (β5) y (β6) se usan (Dv) y (DIg), respectivamente, junto con (I$\neg\neg$C) y (E$\neg\neg$C). ■

Veamos, por fin, que el cálculo que hemos definido es completo, demostrando que las trece condiciones dadas en 7.3 y 7.4 se verifican.

2.-Teorema. *Completitud de* $\mathfrak{CJP}$.

El cálculo de secuencias $\mathfrak{CJP}$ es un cálculo completo para LFP.

Demostración. Las condiciones (1)-(5) de 7.3 son satisfechas inmediatamente usando (Ctd) e (IHip). La condición (6) es el esquema de regla (RFT) y la (7), (IHip).

Para la condición (8) de 7.4 hacemos el siguiente razonamiento:

$\Gamma, \alpha_1, \ldots, \alpha_{m-1}, \alpha_m \vdash \varphi$ (por hipótesis) $\Rightarrow$

$\Rightarrow \Gamma, \alpha_1, \ldots, \alpha_{m-1}, \neg\varphi \vdash \neg\alpha_m$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Gamma, \alpha_1, \ldots, \alpha_{m-1}, \neg\varphi \vdash \neg\alpha$ (por lema 1.(a)) $\Rightarrow$

$\Rightarrow \Gamma, \alpha_1, \ldots, \alpha_{m-1}, \alpha \vdash \varphi$ (por (Ctp_2)) $\Rightarrow$

$\Rightarrow \Gamma, \alpha_1, \ldots, \alpha, \neg\varphi \vdash \neg\alpha_{m-1}$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Gamma, \alpha_1, \ldots, \alpha, \neg\varphi \vdash \neg\alpha$ (por lema 1.(a)) $\Rightarrow$

$\Rightarrow \Gamma, \alpha_1, \ldots, \alpha_{m-2}, \alpha \vdash \varphi$ (por (Ctp_2)) $\Rightarrow$

$\Rightarrow \Gamma, \alpha \vdash \varphi$

repitiendo el proceso anterior.

Para la condición (9):

$\Gamma, \beta_1 \vdash \varphi$ $(1 \leq i \leq m)$ (por hipótesis) $\Rightarrow$

$\Rightarrow \Gamma, \neg\varphi \vdash \neg\beta_1$ $(1 \leq i \leq m)$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Gamma, \neg\varphi \vdash \neg\beta$ (por lema 1.(b)) $\Rightarrow$

$\Rightarrow \Gamma, \beta \vdash \varphi$ (por (Ctp_2)).

Para la condición (10):

$\Gamma, \Delta t \rightarrow \neg\varphi[t/x] \vdash \psi$ (por hipótesis) $\Rightarrow$

$\Rightarrow \Gamma, \neg\Delta t \vee \neg\varphi[t/x] \vdash \psi$ (Definición de $\rightarrow$) $\Rightarrow$

$\Rightarrow \Gamma, \neg\psi \vdash \neg(\neg\Delta t \vee \neg\varphi[t/x])$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Gamma, \neg\psi \vdash \exists x\varphi$ (por $(I\exists C)$ y definición de $\wedge$) $\Rightarrow$

$\Rightarrow \Gamma, \neg\exists x\varphi \vdash \psi$ (por (Ctp_3)).

Para la condición (11):

$\Gamma, \varphi[c/x] \vdash \psi$ (por hipótesis) $\Rightarrow$

$\Rightarrow \Gamma, \neg\psi \vdash \neg\varphi[c/x]$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Gamma, \neg\psi \vdash \neg\exists x\varphi$ (por $(I\forall C)$) $\Rightarrow$

$\Rightarrow \Gamma, \exists x\varphi \vdash \psi$ (por (Ctp_2)).

Para la condición (13) usamos $(I\rightarrow C)$ y $(E\rightarrow C)$. Veamos, por tanto, la condición (12). Por hipótesis tenemos que $\Gamma, \theta \vdash \varphi$ luego $\Gamma \vdash \theta \rightarrow \varphi$, por $(I\rightarrow C)$; por tanto si demostramos que $\Gamma \vdash \theta$ y que $\Gamma \vdash \Delta\theta$, bastará aplicar la regla (MP_Δ) para concluir que $\Gamma \vdash \varphi$.

Demostramos, más en general, que $\vdash \theta$ y $\vdash \Delta\theta$ se verifica para cualquier axioma θ de la igualdad, en el segundo apéndice de este trabajo. ∎

Para concluir esta sección hagamos observar que la lógica que hemos venido describiendo satisface resultados clásicos de las de primer orden. Entre ellos los siguientes teoremas de Löwenheim-Skolem y compacidad.

3.-Teorema.

(a) *Teorema de Löwenheim-Skolem*:

Si un conjunto de fórmulas es fuertemente satisfactible entonces lo es, en una estructura a lo más numerable.

(b) *Teorema de compacidad*:

Si todo subconjunto finito de uno dado Φ es fuertemente satisfactible entonces Φ también lo es.

Demostración. (a) Si Φ es fuertemente satisfactible entonces no tiene un tableau cerrado con lo cual su tableau canónico es abierto y, por tanto, existirá un conjunto de Hintikka H que contiene a Φ y que posee por modelo una estructura a lo más numerable.

(b) Supongamos que Φ no es fuertemente satisfactible entonces resultará que $\Phi \cup \{\neg F\}$ tampoco lo es con lo que $\Phi \vdash F$. Por completitud tendremos entonces que $\Phi \vdash F$, luego existirá un subconjunto finito $\Gamma \subseteq \Phi$ tal que Γ F. En consecuencia $\Gamma \vdash F$, por corrección y, por tanto, Γ no será fuertemente satisfactible. ■

12.- TRABAJOS RELACIONADOS.

Algunos de los autores que han tratado el tema de las funciones parciales en el marco de la lógica de primer orden ya fueron mencionados en el capítulo anterior, cuando se habló de las relaciones de consecuencia lógica que emplearon para hacerlo. Así Hoogewijs [Hoo 83] estudió un cálculo de secuencias y Barringer et. al [BCJ 84], uno de deducción natural -a partir del trabajo del anterior y de Koletsos [Kol 76], sobre el tema. Una comparación de ambos sistemas puede encontrarse en [Hoo 87]. También el citado en el capítulo anterior Owe, presenta en [Owe 84] un sistema formal donde las fórmulas son expresiones booleanas sobre un lenguaje con tipos. Su lógica de primer orden, llamada "weak logic", es presentada a través de otros dos sistemas formales: una lógica base y una extensión de la lógica base. En la primera, el razonamiento es el de la lógica clásica, es decir, $\varphi \vdash \psi$ si φ está definida y si siendo cierta, también lo es ψ. En la segunda, se relaja la condición sobre la hipótesis y únicamente se pide que la conclusión ψ sea cierta si lo era φ. Por último, la "lógica débil" solo exige conclusiones no falsas a partir de hipótesis no falsas. La obtención de un cálculo completo para esta lógica pasa, pues, por la extensión de uno para la lógica clásica de primer orden.

Fueron también citados en el capítulo anterior los trabajos de Urquhart [Urq 86] y Blamey [Bla 86]. El primero es un estudio de las lógicas m-valoradas desde un punto de vista general y trata las funciones parciales en tanto que la divergencia afecta el valor booleano de una fórmula, como se precisó más arriba. Blamey, por el contrario, presenta una lógica parcial para la relación de consecuencia lógica obtenida por la intersección de la ww y la ss-consecuencia, dando para ella un cálculo de secuencias del que se afirma su corrección y completitud.

Ebbinghaus [Ebb 69] presenta también una lógica de primer orden PPL para funciones y predicados parciales en la que las implicaciones siempre están definidas, al estilo de Lukasiewicz. Para ella obtiene un cálculo de secuencias, cuya demostración de completitud se basa en una construcción al estilo de Henkin.

Por último citar el trabajo de Konikowska et al [KTB 88] quienes hacen un estudio de conectivas y cuantificadores 3-valorados antes de especificar las posibles formas de consecuencia lógica. Especificada una de ellas, la ss-consecuencia bajo conectivas de McCarthy y cuantificadores de Kleene -sin

operador de definición- presentan dos cálculos correctos y completos, uno de secuencias y otro de deducción natural. Sin embargo, no se hace mención explícita de las funciones parciales ni de la igualdad.

CAPITULO 3
UNA LOGICA DE PRIMER ORDEN PARA FUNCIONES RECURSIVAS PARCIALES

El sistema formal que acabamos de presentar en el capítulo anterior permite el tratamiento de una sencilla forma de cómputo. Nuestro propósito ahora es ampliar el conjunto de funciones tratadas, teniendo como marco de desarrollo la lógica LFP.

Para ello introducimos un formalismo que permita construirlas. Dicho formalismo es una especie de λ-cálculo que permite definir funciones a través de dos operadores: el habitual de abstracción del λ-cálculo y uno de punto fijo μ, con el que se pueden definir funciones recursivas. Además existirán los símbolos de función de la signatura y las variables funcionales, como nuevos constructores de funciones.

Sobre un conjunto finito de géneros se construye el conjunto de los tipos de manera que cada función, de las que acabamos de citar, posea un tipo. Los mecanismos de construcción de funciones están restringidos de forma que el tipo de toda función es de primer orden, es decir, los tipos que intervienen en su definición, son géneros.

Sobre estos supuestos veremos que es posible dar una semántica sencilla que interprete las funciones como continuas entre cpo's apropiados. La mayor claridad y simplicidad de la aproximación funcional que ofrece el λ-cálculo quedará patente en el siguiente capítulo, cuando se ponga en relación con un lenguaje de programación imperativo.

Con las consideraciones anteriores se construye la lógica para funciones recursivas parciales *LFRP*. En ella existen dos clases de expresiones: las funcionales, obtenidas como acabamos de ver, y las de individuo, obtenidas por aplicación de funciones o por operadores condicionales. Sobre la clase de expresiones se define la de las fórmulas con las mismas características que *LFP*.

La semántica de *LFRP* interpretará, como es lógico, las expresiones funcionales a través de funciones continuas (y estrictas) sobre cpo's planos, definiendo la interpretación del operador μ como el menor punto fijo de una cierta transformación de funciones. Dedicamos las secciones segunda y tercera a demostrar que el operador que acabamos de mencionar es continuo, de forma que el punto fijo exista y la semántica tenga sentido. Tendremos así un

formalismo de carácter puramente funcional.

Tras demostrar algunos resultados semánticos (referidos, entre otros, a la forma de expresar la definición de fórmulas en términos de la de predicados, y a la extensión de la definición a expresiones funcionales), pasamos en la sección quinta a definir las llamadas aproximaciones sintácticas, expresiones que permitirán definir cálculos (no finitarios) para el manejo de funciones definidas por recursión. Con ellas se trata de poder tratar computacionalmente un programa recursivo a través de una regla de punto fijo (cf. [Man 74]). Como veremos entonces, la regla que sustituye toda ocurrencia de una variable funcional por su cuerpo, presenta dificultades técnicas cuando se dan definiciones recursivas anidadas; optaremos pues por una que elimine, en cada aproximación de la función recursiva, todas las apariciones de cualquier otra función recursiva que esté en el alcance de la primera.

El resto de las secciones sigue un desarrollo similar al dado en el capítulo anterior. Se trata, en las primeras, de introducir un método de tableaux apropiado a las peculiaridades del lenguaje que se maneja. Dichas peculiaridades se refieren a los siguientes puntos.

Las clases de fórmulas se amplian con dos nuevos tipos: ω-alfa y ω-beta. Estas nuevas clases no son más que conjunciones y disyunciones infinitas y su comportamiento semántico es análogo al de las clases alfa y beta.

Los tableaux son árboles ω-ramificados, es decir, los nodos pueden tener una cantidad numerable de hijos. A diferencia de los dados para LFP, supondremos, por razones técnicas, que los nodos están etiquetados por conjuntos de fórmulas.

El análogo a los tableaux finitos es ahora el que denominamos tableau de ramas finitas. Se trata de un árbol, posiblemente de profundidad no finita, pero tal que todas sus ramas tienen profundidad finita. La razón de considerar esto así es que con una cantidad numerable de ramas, como se puede llegar a tener, se puede alcanzar una profundidad arbitraria (un ordinal numerable en todo caso).

Las ramas de los tableaux se extienden como en LFP, pero en cada paso puede hacerse con una cantidad arbitraria de ramas. Ello es necesario si esperamos definir un algoritmo que pueda generar un tableau canónico, cerrado en un número finito de pasos.

Como en el capítulo segundo, el método de los tableaux es correcto y completo, con una construcción de tableau canónico, específica, basada en una

técnica "a la Hintikka". La completitud servirá, en la sección décima, para obtener condiciones que demuestran la completitud de cálculos (infinitarios) de secuencias.

Las tres secciones siguientes están dedicadas a exponer un cálculo correcto y completo $\mathfrak{CJR}$, con una regla infinitaria que expresa cuándo una función f, definida a través del operador de recursión μ, no está definida en algún argumento. Dicha regla estará basada en el hecho de que las aproximaciones sintácticas forman una cadena numerable de funciones finitas cuya unión es la función que aproximan. De esta forma, si f no converge en x no lo hará tampoco ninguna de las funciones f^i, $i<\omega$, que la aproximan sintácticamente.

Una lógica infinitaria como ésta, pierde entonces su carácter de primer orden, ya que no satisface, por ejemplo, el teorema de compacidad, aunque sí el de Löwenheim-Skolem, como ocurría con LFP.

1.- LENGUAJE DE UNA LOGICA DE PRIMER ORDEN PARA FUNCIONES RECURSIVAS PARCIALES: LFRP.

El lenguaje que presentamos en este capítulo es una extensión del presentado en el capítulo precedente, que consiste en la generalización de los términos mediante expresiones construidas a la manera de un $\lambda\mu$-cálculo tipado, con las que se pueden definir funciones computables.

En primer lugar, definiremos los tipos que pueden tener las expresiones del lenguaje, que se construirán sobre un conjunto soporte de tipos básicos que se supondrá fijo en adelante.

1.-Definición. *Tipos elementales, relacionales y funcionales.*

Dado un conjunto finito $S = \{\sigma_1,\ldots,\sigma_m\}$, definimos los siguientes conjuntos de tipos:

(1) S, cuyos miembros serán llamados tipos elementales o géneros

(2) TR_S, cuyos miembros ρ serán llamados tipos relacionales, formado por las sucesiones no vacías de tipos elementales $\sigma_1..\sigma_n$, $n>0$, $\sigma_i\in S$

(3) TF_S, cuyos miembros τ serán llamados tipos funcionales y tendrán la forma $\sigma_1..\sigma_n\rightarrow\sigma$, $n>0$, σ_i, $\sigma\in S$. ■

Obsérvese que todo tipo funcional es de la forma $\rho\rightarrow\sigma$, para ciertos $\rho\in TR_S$ y $\sigma\in S$.

Definido el conjunto de los tipos pasamos a definir las signaturas, que habrán de tener en cuenta el hecho de que las expresiones construidas sobre ellas tienen un tipo determinado. Para formalizar esto usaremos el conocido concepto de signatura heterogénea (*many-sorted signature*).

2.-Definición. *Signaturas.*

Una signatura Σ es una cuádrupla formada por:

(1) un conjunto S de géneros $\{\sigma_1,\ldots,\sigma_m\}$

(2) familias de conjuntos C_Σ^σ de constantes c con género asociado σ

(3) familias de conjuntos SF_Σ^τ de símbolos de función f con tipo funcional asociado $\tau=\sigma_1..\sigma_n\rightarrow\sigma$, $n>0$

(4) familias de conjuntos SP_Σ^ρ de símbolos de predicado R con tipo relacional asociado $\rho=\sigma_1..\sigma_n$, $n>0$. ■

En adelante, dada una signatura Σ con un conjunto de géneros S, representaremos por TF_Σ y TR_Σ, los conjuntos de tipos funcionales y relacionales formados a partir de S, respectivamente.

Como para LFP, supondremos dado un conjunto numerable de variables $V^\sigma = \{x_n^\sigma \ / \ n<\omega\}$, que llamaremos de individuo, con un género asociado σ, para cada tipo elemental σ. Asimismo admitiremos variables de tipo funcional, que harán el papel de llamadas a funciones y cuya semántica dependerá del contexto en que aparezcan. Supondremos pues la existencia, para cada $\tau \in TF_\Sigma$, de conjuntos numerables $VF^\tau = \{X_n^\tau \ / \ n<\omega\}$ de variables que llamaremos, simplemente, funcionales. Siempre que no haya ambigüedad eliminaremos subíndice y superíndice en ellas.

3.-<u>Definición</u>. *Σ-términos, Σ-expresiones y Σ-fórmulas.*

Dada una signatura Σ, definimos el conjunto T_Σ^σ de Σ-términos de género $\sigma(\in S)$, el conjunto E_Σ^σ de Σ-expresiones de género $\sigma(\in S)$, el conjunto EF_Σ^τ de Σ-expresiones funcionales de tipo $\tau(\in TF_\Sigma)$ y el conjunto F_Σ de Σ-fórmulas, por las siguientes reglas en forma de Backus-Naur:

(1) $T_\Sigma^\sigma \ni t$:

$t ::= x \ (x \in V^\sigma) \ |$

$c \ (c \in C_\Sigma^\sigma) \ |$

$f(t_1,..,t_n) \ (f \in SF_\Sigma^{\sigma 1..\sigma n \to \sigma}$ y $t_i \in T_\Sigma^{\sigma i}, \ i=1,..,n)$

(2) $E_\Sigma^\sigma \ni E$:

$E ::= t \ (t \in T_\Sigma^\sigma) \ |$

$(\chi \to E_1) \ |$

$(\chi \to E_1, E_2) \ (\chi$ Σ-fórmula sin cuantificadores ni operador de definición Δ y $E_1, E_2 \in E_\Sigma^\sigma) \ |$

$M(E_1,...,E_n) \ (M \in EF_\Sigma^{\sigma 1..\sigma n \to \sigma}$ y $E_i \in E_\Sigma^{\sigma i}, \ i=1,..,n)$

(3) $EF_\Sigma^\tau \ni M$:

$M ::= f \ (f \in SF_\Sigma^\tau) \ |$

$X \ (X \in VF^\tau) \ |$

$\lambda x_1..x_n.E \quad (E \in E_\Sigma^\sigma, \quad x_i \in V^{\sigma i}, \quad i=1,..,n,$ distintas dos a dos, $\tau = \sigma_1..\sigma_n \to \sigma) \ |$

$\mu X.M \ (X \in VF^\tau, \ M \in EF_\Sigma^\tau)$

(4) $F_\Sigma \ni \varphi$:

$\varphi ::= E_1 \simeq_\sigma E_2 \ (E_1,\ E_2 \in E_\Sigma^\sigma) \mid$

$R(E_1,\ldots,E_n) \ \ (R \in SP_\Sigma^{\sigma 1 \ldots \sigma n}$ y $E_i \in E_\Sigma^{\sigma i},\ i=1,\ldots,n) \mid$

$\Delta E \ (E \in E_\Sigma^\sigma) \mid$

$\Delta R(E_1,\ldots,E_n) \ (R \in SP_\Sigma^{\sigma 1 \ldots \sigma n}$ y $E_i \in E_\Sigma^{\sigma i},\ i=1,\ldots,n) \mid$

$\neg \varphi_1 \mid$

$\varphi_1 \vee \varphi_2 \mid$

$\exists x \varphi_1 \ (x \in V^\sigma)$. ■

Varias observaciones hay que hacer sobre esta definición. En primer lugar decir que la sintaxis de las expresiones de tipo funcional incluye las construcciones usuales, para definir funciones, de un $\lambda\mu$-cálculo tipado: la abstracción $\lambda x_1 \ldots x_n . E$, que define una función de n argumentos que obtiene el valor mediante E, y la recursión $\mu X . M$, que define una función como "el menor punto fijo" asociado a M.

En segundo lugar, la sintaxis de las fórmulas admite, como es lógico en primer orden, únicamente la igualdad de tipo elemental. En consecuencia, el operador de definición Δ solo se aplica a ellas. Respecto a la definición de fórmulas, restringimos Δ a predicados con el único objetivo de simplificar nuestro estudio, entendiendo que la ampliación de Δ a fórmulas cualesquiera daría lugar a características lógicas similares a las respectivas de LFP; en todo caso, veremos que la aplicación de Δ a los demás tipos de fórmulas se puede expresar con las construcciones aquí definidas.

Por último hacer notar que, como en el capítulo anterior, el resto de las conectivas y el cuantificador universal, se suponen definidos en función de la negación, la disyunción y el cuantificador existencial.

En adelante, para referirnos a los conjuntos de Σ-términos, Σ-expresiones y Σ-expresiones funcionales, definiremos los conjuntos siguientes:

$$T_\Sigma = \bigcup \{T_\Sigma^\sigma \ / \ \sigma \in S\}$$

$$E_\Sigma = \bigcup \{E_\Sigma^\sigma \ / \ \sigma \in S\}$$

y:

$$EF_\Sigma = \bigcup \{EF_\Sigma^\tau \ / \ \tau \in TF_\Sigma\}.$$

En lo que sigue supondremos dada una signatura fija Σ y escribiremos LFRP para representar nuestro Lenguaje para Funciones Recursivas Parciales formado por el triple $\langle E_\Sigma,\ EF_\Sigma,\ F_\Sigma \rangle$.

2.- SEMANTICA DEL LENGUAJE LFRP. CONCEPTOS INTRODUCTORIOS.

Como hemos visto en la sección anterior, LFRP permite la definición de funciones por recursión, mediante un operador de punto fijo. Por tanto, la semántica debe definirse de tal modo que tales funciones tengan sentido. Para ello dedicamos un breve espacio a explicar las condiciones que permiten la existencia de "menores puntos fijos".

En el capítulo precedente usamos los cpo's planos para definir la semántica de las funciones parciales. Un cpo plano es un caso especial de orden parcial completo, de hecho, quizás el más simple de los no triviales. En general, un cpo se define de la manera siguiente:

1.-Definición. *Ordenes parciales completos.*

Dado un conjunto D con un orden parcial $\leq$, esto es, una relación binaria sobre DxD que sea reflexiva, antisimétrica y transitiva, decimos que dos elementos x, y$\in$D son compatibles, si tienen una extensión común, es decir, existe z$\in$D tal que x, y$\leq$z.

Un subconjunto no vacío de D se dice dirigido si todo par de elementos suyos son compatibles. De entre los conjuntos dirigidos destacamos las cadenas que son aquellos linealmente ordenados, es decir, para cada par de elementos (x, y) de ella, x$\leq$y o y$\leq$x.

Dado un par (D, $\leq$) formado por un conjunto D con un orden parcial $\leq$, decimos que es un orden parcial completo, lo que escribimos con su abreviatura inglesa cpo, si todo conjunto dirigido tiene cota superior mínima y además existe un elemento mínimo, que se representa por $\perp$ y se denomina "bottom" del cpo. ■

Obsérvese que los cpo's planos responden a la definición anterior, pues las cadenas están siempre formadas, a lo más, por el elemento "bottom" y un elemento del conjunto soporte y, por tanto, siempre son cadenas finitas, con lo cual la cota superior mínima es su mayor elemento.

Entre cpo's distinguimos un tipo especial de funciones que permiten definir otras por recursión. Se trata de las funciones continuas, definidas por:

2.-Definición. *Funciones monótonas y continuas.*

Dados los cpo's $(D, \leq)$ y $(D', \leq')$, una función $f: D \longrightarrow D'$ se dice **monótona** si preserva el orden, es decir:

$$x \leq y \Rightarrow f(x) \leq' f(y)$$

para todos $x, y \in D$.

En las mismas condiciones, una función monótona f se dice contínua si preserva las cotas superiores mínimas de cadenas, es decir:

$$f(\bigsqcup \{x_i \;/\; i \in I\}) \leq \bigsqcup \{f(x_i) \;/\; i \in I\}$$

para toda cadena $\{x_i \;/\; i \in I\}$ de D, donde el símbolo $\bigsqcup$ representa la cota superior mínima. Nótese que la monotonía de f garantiza que $\{f(x_i) \;/\; i \in I\}$ sea una cadena. ■

Obsérvese que si f es una función continua entonces, ya que $x_i \leq \bigsqcup\{x_i / i \in I\}$, tendremos que $f(x_i) \leq f(\bigsqcup \{x_i \;/\; i \in I\})$, por ser f monótona, y entonces $\bigsqcup \{f(x_i) \;/\; i \in I\} \leq f(\bigsqcup \{x_i \;/\; i \in I\})$, con lo que:

$$f(\bigsqcup \{x_i \;/\; i \in I\}) = \bigsqcup \{f(x_i) \;/\; i \in I\}.$$

Para definir la semántica de nuestro lenguaje seguiremos las pautas marcadas por LFP y extenderemos la semántica dada en el capítulo anterior a fin de definir correctamente el significado de las expresiones. Mantendremos pues los cpo's planos como base para la obtención de estructuras.

Recuérdese que un cpo plano es una estructura $(D, \leq)$ formada a partir de un conjunto A, un elemento distinguido $\perp$, que se añade a A y se llama "bottom" del cpo, y un orden $\leq$ definido por la condición: $d_1 \leq d_2 \Leftrightarrow d_1 = \perp$ o $d_1 = d_2$. Sobre cpo's planos definimos las siguientes operaciones:

3.-Definición.

(1) Dados los cpo's planos $(D_i, \leq_i)$, $i=1,..,n$, definimos su producto cartesiano estricto $(D_1 \otimes .. \otimes D_n, \leq_\otimes)$ como el orden $\leq_\otimes$ definido sobre el conjunto $D_1 \otimes .. \otimes D_n = \{\perp_\otimes\} \cup \{(a_1,...,a_n) \;/\; a_i \in D_i \setminus \{\perp_i\}\}$ por: $\bar{d}_1 \leq_\otimes \bar{d}_2 \Leftrightarrow \bar{d}_1 = \perp_\otimes$ o $\bar{d}_1 = \bar{d}_2$, para todo $\bar{d}_1, \bar{d}_2 \in D_1 \otimes .. \otimes D_n$.

(2) Dados los cpo's planos $(D, \leq)$ y $(D', \leq')$, definimos el espacio de δ-funciones $([D \longrightarrow D']_\delta, \leq_\delta)$, $\delta \in \{e$ (de estricta), m (de monótona), c (de continua)$\}$, como el orden $\leq_\delta$ definido sobre el conjunto de δ-funciones de D en D', de la siguiente manera. Dados $f, g \in [D \longrightarrow D']_\delta$:

$$f \leq_\delta g \Leftrightarrow f(x) \leq' g(x)$$

para todo $x \in D$. ■

Entiéndase en la definición anterior que una e-función (resp. m o c-función) se refiere a una función estricta (recuérdese que una función estricta entre cpo's es aquella que toma valor "bottom" cuando su argumento es el valor "bottom"; cf. 1.2 del capítulo 2 para la definición de este concepto) (resp. función monótona o continua). Es sencillo demostrar que los espacios de funciones que acabamos de definir son realmente cpo's.

4.-Lema.

Dados los cpo's planos $(D_i, \leq_i)$, $i=1,\ldots,n$, y $(D, \leq)$, se tiene que:

$$[D_1 \bullet \ldots \bullet D_n \to D]_\bullet \subseteq [D_1 \bullet \ldots \bullet D_n \to D]_c.$$

Demostración. Inmediata teniendo en cuenta que, tal como se definió el orden $\leq_\bullet$, las cadenas de $D_1 \bullet \ldots \bullet D_n$ están formadas, a lo más, por $\perp_\bullet$ y una tupla de elementos definidos. ■

Eventualmente usaremos algunas propiedades de las funciones continuas que pasamos a enumerar y cuya demostración puede encontrarse en [Sto 77].

5.-Lema.

Dados los cpo's $(D, \leq)$, $(D', \leq')$ y $(D'', \leq'')$, se tiene:

(a) $id \in [D \to D]_c$, donde $id(x) = x$, para todo $x \in D$

(b) $k_a \in [D \to D']_c$, donde $k_a(x) = a \in D'$, para todo $x \in D$

(c) si $f \in [D \to D']_c$ y $g \in [D' \to D'']_c$ entonces $g \circ f \in [D \to D'']_c$, donde $g \circ f(x) = g(f(x))$, para todo $x \in D$

(d) si f_1, $f_2 \in [D \to D']_c$ y $g \in [D \to \mathbb{B}]_c$, recuérdese que $\mathbb{B}$ es el cpo plano booleano (cf. §1.A.1.1) entonces $cond(g, f_1, f_2) \in [D \to D']_c$, donde $cond(g, f_1, f_2)(x)$ vale $f_1(x)$ si $g(x)$ es $\underline{t}$, $f_2(x)$ si $g(x)$ es $\underline{f}$ y $\perp$, si $g(x)$ es el valor booleano indefinido $\perp$. ■

No trataremos de desarrollar aquí los fundamentos matemáticos de la teoría de dominios, que puede encontrarse en cualquier tratado sobre el tema (cf. [Ber 85] o [Sto 77], por ejemplo). Simplemente daremos cuenta de algunos resultados sobre los que se basa la semántica de nuestro lenguaje.

6.-Teorema. *Teorema de existencia de menor punto fijo.*

Dado un cpo $(D, \leq)$, toda función continua $f: D \to D$ tiene un menor punto fijo, esto es, existe $x \in D$ tal que:

(a) $f(x)=x$

(b) para todo $y\in D$: $f(y)=y \Rightarrow x \leq y$.

Demostración. Consideremos el conjunto $PF=\{f^i(\perp) \ / \ i\in N\}$. Por inducción sobre i demostramos que PF es una cadena. En efecto:

(1) $i=0$: $\perp \leq f^i(\perp)$, por definición de $\perp$

(2) $i\rightarrow i+1$: por hipótesis de inducción $f^i(\perp) \leq f^{i+1}(\perp)$ luego, por monotonía, $f^{i+1}(\perp) \leq f^{i+2}(\perp)$.

Ya que PF es una cadena entonces existe $x = \bigsqcup PF$. Demostremos que x es el menor punto fijo buscado. Tenemos:

(3) $f(x) = f(\bigsqcup \{f^i(\perp) \ / \ i\in I\}) = \bigsqcup \{f(f^i(\perp)) \ / \ i\in I\} = \bigsqcup \{f^i(\perp) \ / \ i\in I\} = x$, usando la continuidad de f.

(4) Supongamos que $f(y)=y$ y demostremos que $f^i(\perp) \leq y$, para todo i, con lo que $x \leq y$, por definición de $\bigsqcup$. Por inducción sobre i:

(a) $i=0$: $\perp \leq y$, por definición de $\perp$

(b) $i\rightarrow i+1$: supongamos que $f^i(\perp) \leq y$ con lo que, por monotonía, $f(f^i(\perp)) = f^{i+1}(\perp) \leq f(y) = y$ (por ser y punto fijo). ■

Como acabamos de ver, las funciones continuas poseen menores puntos fijos. Además el funcional que asocia a cada función continua su menor punto fijo es también continuo, como vemos a continuación.

7.-Teorema. *Continuidad del operador de punto fijo.*

Dado un cpo $(D, \leq)$, el funcional $fix: [D\rightarrow D]_c \rightarrow D$ definido por:

$$fix(f) = \text{menor punto fijo de } f$$

para toda $f\in[D\rightarrow D]_c$, es continuo.

Demostración. Para demostrar que fix es monótono obsérvese que el menor punto fijo de una función continua f es de la forma $\bigsqcup \{f^i(\perp) \ / \ i\in I\}$, de donde se sigue fácilmente que $fix(f) \leq fix(g)$ si $f \leq g$. Para la continuidad es necesario que para toda cadena $\{f_i \ / \ i\in I\}\subseteq[D\rightarrow D]_c$ se tenga:

$$fix(\bigsqcup_i f_i) \leq \bigsqcup_i fix(f_i).$$

Para ello basta que $\bigsqcup_i fix(f_i)$ sea punto fijo de $\bigsqcup_i f_i$, ya que el menor de ellos es $fix(\bigsqcup_i f_i)$. Ahora bien, se tiene:

$$(\bigsqcup_i f_i)(\bigsqcup_j fix(f_j)) = \bigsqcup_i (f_i(\bigsqcup_j fix(f_j))) \text{ (por definición de } \bigsqcup_i f_i) =$$

$$= \bigsqcup_i \bigsqcup_j (f_i(fix(f_j))) \text{ (por continuidad de } f_i) =$$

$= \bigcup_j (f_j(\mathrm{fix}(f_j)))$ (comprobable usando la definición de supremo; para más detalles cf. [Bak 80]) $=$

$= \bigcup_j \mathrm{fix}(f_j)$ (por definición de punto fijo). ■

3.- SEMANTICA DE LFRP.

Como hemos dicho en la sección precedente, mantendremos las ideas del capítulo segundo y expresaremos las funciones parciales como funciones estrictas sobre cpo's planos, reservando su elemento base para la divergencia. Las Σ-estructuras serán pues similares a las definidas entonces, con ciertas modificaciones técnicas a tener en cuenta cuando se trata con signaturas heterogéneas.

1.-Definición. *Σ-estructuras.*

Una Σ-estructura $\mathfrak{A}$ es una cuádrupla formada por una familia de conjuntos $\{D^\sigma \ / \ \sigma \in S\}$, donde cada D^σ es un cpo plano sobre un conjunto $A^\sigma \cup \{\perp_\sigma\}$ (unión disjunta), donde $\perp_\sigma$ es el elemento mínimo y A^σ será llamado soporte del género σ en $\mathfrak{A}$; un conjunto de elementos $c^{\mathfrak{A}} \in D^\sigma$, uno por cada constante $c \in C_\Sigma^\sigma$, un conjunto de funciones estrictas $f^{\mathfrak{A}} \in [D^{\sigma_1} \otimes .. \otimes D^{\sigma_n} \to D^\sigma]_\bullet$, una por cada símbolo de función $f \in SF_\Sigma^\tau$ de tipo funcional $\tau = \sigma_1 .. \sigma_n \to \sigma$, y un conjunto de funciones estrictas $R^{\mathfrak{A}} \in [D^{\sigma_1} \otimes .. \otimes D^{\sigma_n} \to \mathbb{B}]_\bullet$, una por cada símbolo de predicado $R \in SP_\Sigma^\rho$ de tipo relacional $\rho = \sigma_1 .. \sigma_n$. ■

Las valoraciones se extienden ahora para cubrir también las variables funcionales, manteniendo la idea de que ellas representan funciones estrictas.

2.-Definición. *Valoraciones y Σ-interpretaciones.*

Dada una Σ-estructura $\mathfrak{A}$, una valoración ε sobre $\mathfrak{A}$ es una aplicación definida sobre todos los conjuntos de variables tal que:

(1) $\varepsilon(x^\sigma) \in A^\sigma$ $(= D^\sigma \setminus \{\perp_\sigma\})$

(2) $\varepsilon(X^\tau) \in [D^{\sigma_1} \otimes .. \otimes D^{\sigma_n} \to D^\sigma]_\bullet$, $\tau = \sigma_1 .. \sigma_n \to \sigma$, $\tau \in TF_\Sigma$.

Una Σ-interpretación $\mathfrak{I}$ es un par $(\mathfrak{A}, \varepsilon)$ formado por una Σ-estructura $\mathfrak{A}$ y una valoración ε sobre $\mathfrak{A}$. ■

Obsérvese que, de nuevo, las variables de individuo siempre se valoran como elementos definidos.

La semántica de expresiones y fórmulas -la de los términos es la misma que la dada en §2.1.5.(1)- puede ahora definirse de manera similar a la de LFP. La justificación de la existencia de ciertos puntos fijos mínimos se hará después de esta definición.

3.-Definición. *Valores semánticos de expresiones y fórmulas.*

Dada una Σ-interpretación $\mathfrak{I}$, la semántica de expresiones y fórmulas viene dada por aplicaciones que asignan a cada $E \in E_\Sigma^\sigma$, un elemento $\mathfrak{I}(E) \in D^\sigma$, a cada $M \in EF_\Sigma^\tau$, $\tau = \sigma_1 .. \sigma_n \to \sigma$, una función estricta $\mathfrak{I}(M) \in [D^{\sigma_1} \otimes .. \otimes D^{\sigma_n} \to D^\sigma]_e$, y a cada fórmula $\varphi \in F_\Sigma$, un valor booleano $\mathfrak{I}(\varphi) \in B$, de la manera siguiente:

(1) $E_\Sigma^\sigma \ni E$:

$\mathfrak{I}(E) = \mathfrak{I}(t)$, si $E = t (\in T_\Sigma^\sigma)$ (cf. §2.1.5.(1))

$$\mathfrak{I}(\chi \to E_1, E_2) = \begin{cases} \mathfrak{I}(E_1) & \text{si } \mathfrak{I}(\chi) = \underline{t} \\ \mathfrak{I}(E_2) & \text{si } \mathfrak{I}(\chi) = \underline{f} \\ \perp_\sigma & \text{en otro caso} \end{cases}$$

Análogamente $\mathfrak{I}(\chi \to E_1)$.

$\mathfrak{I}(M(E_1, .., E_n)) = \mathfrak{I}(M)(\mathfrak{I}(E_1), .., \mathfrak{I}(E_n))$

En adelante, entenderemos $(\ , .., \ )$ como una operación estricta de construcción de tuplas, esto es, si alguna de las componentes de la misma está indefinida entonces la tupla es el "bottom" del producto cartesiano estricto. Recuérdese que dicho producto solo tenía como tuplas, distintas del "bottom", a aquellas cuyas componentes eran todas distintas del elemento mínimo del correspondiente miembro del producto.

(2) $EF_\Sigma^\tau \ni M$, $\tau = \sigma_1 .. \sigma_n \to \sigma$:

$\mathfrak{I}(f) = f^{\mathfrak{A}}$

$\mathfrak{I}(X^\tau) = \varepsilon(X^\tau)$

$\mathfrak{I}(\lambda x_1 .. x_n . E)$ es la función estricta de $[D^{\sigma_1} \otimes .. \otimes D^{\sigma_n} \to D^\sigma]_e$ definida por:

$\mathfrak{I}(\lambda x_1 .. x_n . E)(a_1, .., a_n) = \mathfrak{I}[a_1/x_1, .., a_n/x_n](E)$

para $a_i \in A^{\sigma_i}$, $i=1,\ldots,n$, donde esta última interpretación se define de manera obvia a partir de la dada en §2.1.5.(2).

$\mathcal{J}(\mu X.M)$ es el menor punto fijo de la transformación que asocia a cada $h \in [D^{\sigma_1}\bullet\ldots\bullet D^{\sigma_n}\to D^{\sigma}]_\bullet$, la función estricta $\mathcal{J}[h/X](M)$.

(3) Para F_Σ, la definición es la misma que la dada en el capítulo anterior para LFP, cuando la fórmula es una negación o una disyunción. Para el resto de los casos se tiene:

$$\mathcal{J}(E \simeq_\sigma E') = \begin{cases} \underline{t} & \text{si } \perp_\sigma \neq \mathcal{J}(E) = \mathcal{J}(E') \neq \perp_\sigma \\ \underline{f} & \text{si } \perp_\sigma \neq \mathcal{J}(E) \neq \mathcal{J}(E') \neq \perp_\sigma \\ \perp & \text{en otro caso} \end{cases}$$

$$\mathcal{J}(R(E_1,\ldots,E_n)) = \begin{cases} \underline{t} & \text{si } R^{\mathfrak{A}}(\mathcal{J}(E_1),\ldots,\mathcal{J}(E_n)) = \underline{t} \\ \underline{f} & \text{si } R^{\mathfrak{A}}(\mathcal{J}(E_1),\ldots,\mathcal{J}(E_n)) = \underline{f} \\ \perp & \text{en otro caso} \end{cases}$$

$$\mathcal{J}(\Delta E) = \begin{cases} \underline{t} & \text{si } \mathcal{J}(E) \neq \perp_\sigma \\ \underline{f} & \text{en otro caso} \end{cases}$$

$$\mathcal{J}(\Delta R(E_1,\ldots,E_n)) = \begin{cases} \underline{t} & \text{si } \mathcal{J}(R(E_1,\ldots,E_n)) \neq \perp \\ \underline{f} & \text{en otro caso} \end{cases}$$

$$\mathcal{J}(\exists x^\sigma \varphi) = \begin{cases} \underline{t} & \text{si existe } a \in A^\sigma \text{ tal que } \mathcal{J}[a/x^\sigma](\varphi) = \underline{t} \\ \underline{f} & \text{si } \mathcal{J}[a/x^\sigma](\varphi) = \underline{f} \text{ para todo } a \in A^\sigma \\ \perp & \text{en otro caso} \end{cases}$$

■

Para que la definición anterior tenga sentido hemos de demostrar, entre otras cosas, que $\mathcal{J}(M)$ queda bien definida y es una función estricta, para $M \in EF^\tau_\Sigma$, $\tau=\sigma_1\ldots\sigma_n\to\sigma$.

4.-<u>Teorema</u>.

Dados una interpretación $\mathcal{J}$, una variable funcional $X \in VF^\tau$, una expresión $E \in E^\sigma_\Sigma$ de género σ, una expresión funcional $M \in EF^\tau_\Sigma$ y una fórmula $\chi \in F_\Sigma$ libre de cuantificadores y de operador de definición Δ:

(a) $T(\mathcal{J}, X, E)$: $[D^{\sigma_1}\bullet\ldots\bullet D^{\sigma_n}\to D^{\sigma}]_\bullet \to D^\sigma$ definida por:

$$T(\mathcal{J}, X, E)(h) = \mathcal{J}[h/X](E)$$

satisface:

(i) T(J, X, E) está bien definida

(ii) T(J, X, E) es continua

(b) T(J, X, χ): $[D^{\sigma_1}\oplus..\oplus D^{\sigma_n}\to D^{\sigma}]_e \to B$ definida por:

T(J, X, χ)(h) = J[h/X](χ)

satisface:

(i) T(J, X, χ) está bien definida

(ii) T(J, X, χ) es continua

(c) T(J, X, M): $[D^{\sigma_1}\oplus..\oplus D^{\sigma_n}\to D^{\sigma}]_e \to [D^{\sigma_1}\oplus..\oplus D^{\sigma_n}\to D^{\sigma}]_e$ definida por:

T(J, X, M)(h) = J[h/X](M)

satisface:

(i) T(J, X, M) está bien definida

(ii) T(J, X, M) es continua.

Demostración. Se demuestran los tres apartados por inducción simultánea sobre la estructura de E, χ y M. Tenemos:

(a) Que T(J, X, E) está bien definida se comprueba inmediatamente, distinguiendo las diferentes formas posibles para E. Por ejemplo, si E es de la forma $M(E_1,...,E_n)$ entonces, para cualquier interpretación J', $J'(M)\in[D^{\sigma_1}\oplus..\oplus D^{\sigma_n}\to D^{\sigma}]_e$ e $J'(E_i)\in D^{\sigma_i}$, i=1,..,n, por las hipótesis de inducción para M y las E_i, luego $J'(E)\in D^{\sigma}$.

Respecto a la continuidad, obsérvese que para E=t se puede demostrar, por inducción sobre la estructura de t, que T(J,X,t) es constante luego, por 2.5.(b), continua. Para $E=\chi\to E_1$, E_2, análogamente $\chi\to E_1$, aplicamos hipótesis de inducción para T(J, X, E_i), i=1,2, y T(J, X, χ) y usamos 2.5.(d). Por último, para $E=M(E_1,...,E_n)$ basta usar la continuidad de las funciones T(J, X, M) y T(J, X, E_i), i=1,..,n, que nos da la hipótesis de inducción, y aplicar que ésta se conserva bajo composición, según 2.5.(c).

(b) La buena definición de T(J, X, χ) es inmediata. Respecto a su continuidad distinguimos los siguientes casos. Para $E\simeq_\sigma E'$ tenemos:

$J[\bigsqcup_i h_i/X](E\simeq_\sigma E') = \underline{t} \Leftrightarrow \perp_\sigma \neq J[\bigsqcup_i h_i/X](E) = J[\bigsqcup_i h_i/X](E') \neq \perp_\sigma \Leftrightarrow$

$\Leftrightarrow \perp_\sigma \neq \bigsqcup_i (J[h_i/X](E)) = \bigsqcup_i (J[h_i/X](E')) \neq \perp_\sigma$ (por hip. de inducción) $\Leftrightarrow$

$\Leftrightarrow J[h_i/X](E) = J[h_i/X](E') \neq \perp_\sigma$, para algún i (por tratarse de cpo's planos) $\Leftrightarrow$

$\Leftrightarrow J[h_i/X](E\simeq_\sigma E') = \underline{t}$, para algún i $\Leftrightarrow \bigsqcup_i (J[h_i/X](E\simeq_\sigma E')) = \underline{t}$

Similarmente si la igualdad es falsa o indefinida (en este caso recuérdese que la igualdad se interpreta como una función estricta). De manera análoga se demuestra la continuidad para los predicados (recordando de

nuevo que $R^{\mathfrak{A}}$ es una función estricta). Por hipótesis de inducción resultará, entonces, la continuidad para la negación y la disyunción.

(c) La buena definición de T(J, X, M) es inmediata para las expresiones funcionales f, X y $\lambda x_1..x_n.E$, por la definición semántica 3.(2) y la hipótesis de inducción para E. La continuidad, en estos casos, es fácil de demostrar. Para T(J, X, f) y T(J, X, Y), Y≠X, resulta una función constante y, por tanto, continua, por 2.5.(b). Para T(J, X, X) resulta la función identidad que es continua, por 2.5.(a). Para T(J, X, $\lambda x_1..x_n.E$) resulta:

$$(J[\bigcup_i h_i/X](\lambda x_1..x_n.E))(a_1,..,a_n) = J[\bigcup_i h_i/X][a_1/x_1,...,a_n/x_n](E) =$$

$$= J[a_1/x_1,...,a_n/x_n][\bigcup_i h_i/X](E) = T(J', X, E)(\bigcup_i h_i)$$

(con $J' = J[a_1/x_1,...,a_n/x_n]$) =

$$= \bigcup_i (T(J', X, E)(h_i)) \text{ (por hipótesis de inducción para E)} =$$

$$= \bigcup_i (J[a_1/x_1,...,a_n/x_n][h_i/X](E)) = \bigcup_i (J[h_i/X][a_1/x_1,...,a_n/x_n](E)) =$$

$$= \bigcup_i ((J[h_i/X](\lambda x_1..x_n.E))(a_1,..,a_n)) =$$

$$= (\bigcup_i (J[h_i/X](\lambda x_1..x_n.E)))\ (a_1,...,a_n)$$

luego T(J, X, $\lambda x_1..x_n.E$) es continua.

Respecto a $\mu Y.M'$, Y≠X, tenemos que:

$T(J, X, \mu Y.M')(h) = J[h/X](\mu Y.M')$ (por def. de T) = $fix(T(J[h/X], Y, M'))$

con lo que $T(J, X, \mu Y.M')(h)$ está bien definida, por hipótesis de inducción para M'. Para la continuidad, consideremos el funcional definido por:

$$F(h) = T(J[h/X], Y, M').$$

Si demostramos su continuidad entonces, como $T(J, X, \mu Y.M')(h) = fix(T(J[h/X], Y, M') = fix(F(h))$, resultará que $T(J, X, \mu Y.M')$ es continua, por ser composición de dos funciones continuas. Ahora bien, para cualquier cadena $\{h_i / i\in I\} \subseteq [D^{\sigma_1}e..eD^{\sigma_n}\to D^{\sigma}]_e$ y cualquier función $f\in[D^{\sigma_1}e..eD^{\sigma_n}\to D^{\sigma}]_e$ tenemos:

$$T(J[\bigcup_i h_i/X], Y, M')(f) = J[\bigcup_i h_i/X][f/Y](M') =$$

$$= J[f/Y][\bigcup_i h_i/X](M') \text{ (pues } X\neq Y) = T(J[f/Y], X, M')(\bigcup_i h_i) =$$

$$= \bigcup_i T(J[f/Y], X, M')(h_i) \text{ (por continuidad de } T(J[f/Y], X, M'), \text{ por}$$

hipótesis de inducción) =

$$= \bigcup_i J[f/Y][h_i/X](M') = \bigcup_i J[h_i/X][f/Y](M') \text{ (pues } X\neq Y) =$$

$= \bigcup_i T(\mathfrak{I}[h_i/X],\ Y,\ M')(f)$

luego $T(\mathfrak{I}[\bigcup_i h_i/X],\ Y,\ M') = \bigcup_i T(\mathfrak{I}[h_i/X],\ Y,\ M')$, que prueba la continuidad de F.

Si Y=X entonces $T(\mathfrak{I},\ X,\ \mu X.M)$ está bien definido, por hipótesis de inducción, y por tratarse además de un funcional constante, su continuidad es trivial. ■

Demostrado que $T(\mathfrak{I},\ X,\ M)$ es continua, por 2.6 tendrá punto fijo y, por tanto, $\mathfrak{I}(M)$ estará bien definida, para cualquier expresión funcional M.

La razón de que exijamos que la fórmula χ no contenga ninguna aparición del operador de definición Δ es que en otro caso la función $T(\mathfrak{I},\ X,\ \chi)$ no es siquiera monótona en general. Intuitivamente cabe esperar esto si recordamos que Δ no es un operador monótono, pues para $\mathfrak{I}$, $\mathfrak{I}'$ y φ adecuadas puede suceder que $\mathfrak{I}(\varphi)=\bot \leq \mathfrak{I}'(\varphi)=\underline{t}$, mientras que $\mathfrak{I}(\Delta\varphi)=\underline{f}$ no es menor que $\mathfrak{I}'(\Delta\varphi)=\underline{t}$. Veamos un ejemplo concreto considerando la fórmula $\Delta X^{\tau}(x_1,\ldots,x_n)$, $\tau=\sigma_1..\sigma_n\rightarrow\sigma$. Se tiene:

$$\mathfrak{I}[\Omega/X](\Delta X^{\tau}(x_1,\ldots,x_n)) = \underline{f}$$

y:

$$\mathfrak{I}[k_a/X](\Delta X^{\tau}(x_1,..,x_n)) = \underline{t}$$

donde Ω, $k_a \in [D^{\sigma_1}\otimes..\otimes D^{\sigma_n}\rightarrow D^{\sigma}]_e$ son las funciones estrictas definidas por:

$$\Omega(a_1,\ldots,a_n) = \bot_{\sigma} \text{ y } k_a(a_1,\ldots,a_n) = a(\in A^{\sigma})$$

para toda n-tupla $(a_1,\ldots,a_n)\in D^{\sigma_1}\otimes..\otimes D^{\sigma_n} \setminus \{\bot_{\otimes}\}$.

Entonces se tiene que $\Omega \leq k_a$, pues $\bot_{\sigma} \leq a$, pero no es cierto que $\underline{f}$ sea menor que $\underline{t}$.

Una vez demostrado que la semántica de nuestro lenguaje está bien definida usaremos el hecho de que podemos caracterizar los puntos fijos mínimos de una manera concreta, según se vio en 2.6.

5.-Corolario.

Dadas una Σ-interpretación $\mathfrak{I}$, una expresión funcional $M\in EF_{\Sigma}^{\tau}$ y una variable funcional $X\in VF^{\tau}$, $\tau=\sigma_1..\sigma_n\rightarrow\sigma$, definamos la siguiente sucesión de funciones $\{<\mu X.M>^i \in [D^{\sigma_1}\otimes..\otimes D^{\sigma_n}\rightarrow D^{\sigma}]_e \ / \ i<\omega\}$, por inducción sobre i:

$$<\mu X.M>^0 = \Omega_{\tau} \text{ (la función } \Omega \text{ definida más arriba)}$$

$$<\mu X.M>^{i+1} = \mathfrak{I}[<\mu X.M>^i /X](M).$$

Entonces $\mathcal{I}(\mu X.M) = \bigcup_i \{<\mu X.M>^i \ / \ i<\omega\}$.

Demostración. En efecto, por 2.6 tenemos que:

$$\mathcal{I}(\mu X.M) = \bigcup_i \{T(\mathcal{I}, X, M)^i(\perp_\tau) \ / \ i<\omega\}$$

Ahora bien $<\mu X.M>^i = T(\mathcal{I}, X, M)^i(\perp_\tau)$. En efecto, por inducción sobre i tenemos que:

i=0. Es evidente pues $\perp_\tau = \Omega_\tau$.

$i \longrightarrow i+1$. Supongámoslo cierto para i entonces:

$$<\mu X.M>^{i+1} = \mathcal{I}[<\mu X.M>^i/X](M) =$$
$$= \mathcal{I}[T(\mathcal{I}, X, M)^i(\perp_\tau)/X](M) \text{ (por hip. de ind.)}$$
$$= T(\mathcal{I}, X, M)(T(\mathcal{I}, X, M)^i(\perp_\tau)) = T(\mathcal{I}, X, M)^{i+1}(\perp_\tau)$$

El corolario resulta ahora de forma inmediata. ■

4.- ALGUNOS RESULTADOS SEMANTICOS.

De la semántica que acabamos de definir se deducen algunos resultados que pasamos a comentar a continuación. En primer lugar, vale lo dicho en el capítulo anterior sobre las fórmulas siempre ciertas Δx y siempre falsas $\neg\Delta x$, para cualquier variable de individuo x. Escribiremos, sin embargo, T (resp. F) para representar $x_0 \simeq x_0$ (resp. $\neg(x_0 \simeq x_0)$), siendo x_0, la primera variable de un género arbitrario fijo. Evitamos el uso del operador de definición Δ y posibilitamos así que dichas fórmulas aparezcan en expresiones condicionales. Igualmente escribiremos ω para la expresión siempre indefinida $F \longrightarrow x_0$. Con dicha expresión es posible definir una función que siempre tome valores indefinidos. En efecto, dado $\tau=\sigma_1..\sigma_n \longrightarrow \sigma$, sea M la expresión $\lambda x_1..x_n.\neg(x_1 \simeq x_1) \longrightarrow x_0$, donde $x_i \in V^{\sigma_i}$, i=1,...,n, y $x_0 \in V^\sigma$. Para cualquier interpretación $\mathcal{I}$ y tupla $(a_1,...,a_n) \in D^{\sigma_1} \otimes .. \otimes D^{\sigma_n} \setminus \{\perp_\otimes\}$ se tiene $\mathcal{I}(M)(a_1,...,a_n) = \mathcal{I}[a_1/x_1,...,a_n/x_n](\neg(x_1 \simeq x_1) \longrightarrow x_0) = \perp_\sigma$ (pues siempre $a_1 = a_1$). Por tanto $\mathcal{I}(M) = \Omega_\tau$. Para evitar complicar la notación escribiremos Ω_τ para referirnos tanto a la expresión que acabamos de definir como a su denotación.

Es posible definir una función siempre indefinida sin hacer uso de expresiones condicionales. Para ello consideremos la variable funcional $X \in VF^\tau$, $\tau=\sigma_1..\sigma_n \longrightarrow \sigma$ y las variables de individuo $x_i \in V^{\sigma_i}$, i=1,..,n. Sea la expresión $\mu X.\lambda x_1..x_n.X(x_1,...,x_n)$ y consideremos una Σ-interpretación $\mathcal{I}$ cualquiera. Entonces, según 3.5:

$$\mathfrak{I}(\mu X.\lambda x_1..x_n.X(x_1,...,x_n)) = \bigcup_i \{\mathfrak{I}(<\mu X.M>^i) \;/\; i<\omega\}$$

donde $<\mu X.M>^i$ se define inductivamente como sigue:

$$<\mu X.M>^0 = \Omega_\tau$$

$$<\mu X.M>^{i+1} = \mathfrak{I}[<\mu X.M>^i/X](\lambda x_1..x_n.X(x_1,...,x_n))$$

Demostremos por inducción sobre i que $<\mu X.M>^i = \Omega_\tau$, para todo $i<\omega$. Para i=0 es evidente. Supuesto para i y dados $a_j \in D^{\sigma_j}$, j=1,...,n, se tiene:

$$<\mu X.M>^{i+1}(a_1,...,a_n) = \mathfrak{I}[<\mu X.M>^i/X][a_1/x_1,...,a_n/x_n](X(x_1,...,x_n)) =$$

$$= <\mu X.M>^i(a_1,...,a_n) = \perp_\sigma \text{ (por hipótesis de inducción)}$$

Por tanto:

$$\mathfrak{I}(\mu X.\lambda x_1..x_n.X(x_1,...,x_n)) = \Omega_\tau$$

En cualquier caso, la fórmula $\omega=x_0$ siempre estará indefinida, por estarlo el miembro izquierdo de la igualdad.

Por otra parte, como hemos visto, el operador de definición Δ solo se aplica a predicados ya que, según se dijo, el resto es expresable en función de estos. Veamos que, en efecto, esto es así.

1.-Definición. *Definición de fórmulas.*

El operador de definición Δ aplicado a una fórmula φ, escrito $\Delta\varphi$, se define por inducción sobre la estructura de φ mediante:

$\Delta R(E_1,...,E_n)$ está ya entre las fórmulas

$\Delta(E_1 \propto E_2) = \Delta E_1 \wedge \Delta E_2$

$\Delta\neg\varphi = \Delta\varphi$

$\Delta(\varphi\vee\psi) = (\Delta\varphi\vee\psi) \wedge (\varphi\vee\Delta\psi) \wedge (\Delta\varphi\vee\Delta\psi)$

$\Delta\exists x\varphi = \forall x\Delta\varphi \vee \exists x(\Delta\varphi\wedge\varphi)$

$\Delta\Delta\varphi = T$

$\Delta\Delta E = T.$ ■

Se puede demostrar fácilmente que $\Delta\varphi$ se comporta como cabe esperar, esto es, que es cierta si φ no es indefinida, y es falsa en otro caso. Para ello basta seguir la demostración de §2.3.2.

También se puede extender la aplicación de Δ a expresiones de tipo funcional. En efecto, si entendemos que una función está definida cuando lo está en algún argumento entonces:

$$\Delta M = \exists x_1..\exists x_n \Delta M(x_1,...,x_n)$$

para toda $M \in EF^\tau_\Sigma$, $\tau=\sigma_1..\sigma_n \longrightarrow \sigma$.

También la igualdad de dos funciones se puede expresar en el lenguaje. En principio, caben dos formas de entender la igualdad, como ocurría con la igualdad débil y fuerte de términos en LFP (cf. §2.1). Una, la fuerte $\equiv$, es la que se hace cierta cuando para cualquier tupla de argumentos, si una de las funciones converge entonces la otra también y ambas lo hacen al mismo valor, y si no converge, la otra tampoco; y se hace falsa en otro caso. Es decir, dadas dos expresiones M, $M' \in EF^{\tau}_{\Sigma}$, $\tau = \sigma_1..\sigma_n \rightarrow \sigma$, se tiene:

$$(M \equiv M') = \forall x_1..\forall x_n [(\Delta M(x_1,..,x_n) \wedge \Delta M'(x_1,..,x_n) \wedge M(x_1,..,x_n) \simeq M'(x_1,..,x_n)) \vee (\neg\Delta M(x_1,..,x_n) \wedge \neg\Delta M'(x_1,..,x_n))]$$

Obsérvese que la fórmula anterior solo puede llegar a tener valor booleano cierto o falso. Es por esto, en este sentido, una igualdad fuerte (compárese con la igualdad fuerte definida en §2.1). La correspondiente igualdad débil sería simplemente la fórmula:

$$(M \simeq M') = \forall x_1..\forall x_n (M(x_1,..,x_n) \simeq M'(x_1,..,x_n))$$

y podría tomar los tres valores booleanos. Para que se hiciera cierta las dos funciones habrían de estar definidas y tomar el mismo valor, sobre todos los argumentos.

Los conceptos de aparición libre de una variable en una expresión o una fórmula se definen extendiendo los ya dados en §2.2.1, para tener en cuenta también las variables funcionales.

2.-<u>Definición</u>. *Variables (libres) en expresiones y fórmulas.*

Dadas una expresión E y una fórmula φ, se definen los conjuntos de apariciones de variables (resp. libres) en E, escrito var(E) (resp. lib(E)), y en φ, escrito var(φ) (resp. lib(φ)), por inducción sobre la estructura de E y φ, como sigue:

(1) $E^{\sigma}_{\Sigma} \ni E$:

var(t) se define como en §2.2.1 y lib(t) = var(t)

$var(\chi \rightarrow E) = var(\chi) \cup var(E)$

$var(\chi \rightarrow E, E') = var(\chi) \cup var(E) \cup var(E')$

$var(M(E_1,..,E_n)) = var(M) \cup \bigcup \{var(E_i) \;/\; i=1,..,n\}$ y análogamente para las libres, cambiando var por lib.

(2) $EF^{\tau}_{\Sigma} \ni M$, $\tau = \sigma_1..\sigma_n \rightarrow \sigma$:

$var(f) = lib(f) = \emptyset$

$var(X) = lib(X) = \{X\}$

$var(\lambda x_1..x_n.E) = var(E) \cup \{x_1,...,x_n\}$

$lib(\lambda x_1..x_n.E) = lib(E) \setminus \{x_1,...,x_n\}$

$var(\mu X.M) = var(M) \cup \{X\}$

$lib(\mu X.M) = lib(M) \setminus \{X\}$

(3) Para fórmulas tomamos la definición de §2.2.1. ■

Como en el capítulo anterior, puede demostrarse un resultado de coincidencia al estilo siguiente.

3.-Lema. *Coincidencia.*

Dadas dos Σ-interpretaciones $\mathfrak{I}$ e $\mathfrak{I}'$ cualesquiera, ambas con el mismo dominio, se tiene que si $\mathfrak{I}$ e $\mathfrak{I}'$ interpretan de la misma forma los símbolos de Σ entonces, para toda expresión $E \in E_\Sigma \cup EF_\Sigma$ (resp. $\varphi \in F_\Sigma$), si $\mathfrak{I}(x) = \mathfrak{I}'(x)$ para toda variable de individuo o funcional $x \in lib(E)$ (resp. $lib(\varphi)$) entonces $\mathfrak{I}(E) = \mathfrak{I}'(E)$ (resp. $\mathfrak{I}(\varphi) = \mathfrak{I}'(\varphi)$). ■

Análogamente puede definirse el concepto de sustitución de una variable (del tipo que sea) por una expresión (del mismo tipo) en otra expresión o en una fórmula.

4.-Definición. *Sustitución.*

Dada una variable x y una expresión E ambas del mismo tipo, se define la sustitución de x por E en una expresión E', escrito $E'[E/x]$, o una fórmula φ, escrito $\varphi[E/x]$, por inducción sobre la estructura de E' y φ como sigue:

(1) $E \in E_\Sigma^\sigma$. La sustitución en términos se define como en §2.2.3.(1). Para las expresiones condicionales se define análogamente a las *if* χ *then* t e *if* χ *then* t *else* t' del capítulo anterior. Por último:

$$(M(E_1,...,E_n))[E/x] = M[E/x](E_1[E/x],...,E_n[E/x])$$

(2) $M \in EF_\Sigma^\tau$, $\tau = \sigma_1..\sigma_n \to \sigma$. Para f o X, la definición es inmediata. Para el resto tenemos:

$$(\lambda x_1..x_n.E')[E/x] = \begin{cases} \lambda x_1..x_n.E'[E/x] & \text{si } x \neq x_i,\ i=1,..,n,\ \text{y} \\ & \quad \text{y } lib(E) \cap \{x_1,...,x_n\} = \emptyset \\ \lambda..z_1..z_m...E'[z_1/x_{i_1},...,z_m/x_{i_m}][E/x] & \text{si } x \neq x_i \\ & \quad \text{y } lib(E) \cap \{x_1,...,x_n\} = \{x_{i_1},...,x_{i_m}\},\ m>0 \\ \lambda x_1..x_n.E' & \text{en otro caso} \end{cases}$$

donde "$\lambda..z_i..z_m..$" abrevia "$\lambda x_1..x_{i1-1}z_1x_{i1+1}..x_{im-1}z_mx_{im+1}..x_n$" y donde z_j, $x_{ij}\in V^{\sigma_j}$, $j=1,...,m$, y cada z_j es la primera variable tal que $z_j\notin\{x_1,...,x_n\}\cup lib(E)\cup lib(E')\cup\{x\}$, $j=1,..,m$.

Análogamente se trata la colisión de variables en expresiones como $\mu X.M$.

(3) Para fórmulas tomamos la definición de §2.2.3.(2). ∎

Se puede demostrar un resultado sobre la sustitución análogo al dado en §2.2.4.

5.-Lema. *Sustitución.*

Dadas una Σ-interpretación $\mathfrak{I}$, una expresión $E'\in E_\Sigma\cup EF_\Sigma$ y una fórmula $\varphi\in F_\Sigma$ se tiene:

(a) $\mathfrak{I}(E'[E/x]) = \mathfrak{I}[\mathfrak{I}(E)/x](E')$

(b) $\mathfrak{I}(\varphi[E/x]) = \mathfrak{I}[\mathfrak{I}(E)/x](\varphi)$

supuesto que E y x tengan el mismo tipo e $\mathfrak{I}(E)\neq i_\sigma$, si $x\in V^\sigma$.

Demostración. Unicamente demostraremos los casos nuevos, siendo análoga a LFP, la demostración del resto. De los nuevos, veremos como caso típico la abstracción, siendo similar la recursión. Distinguimos, como en la definición:

(i) $x\notin\{x_1,...,x_n\}$ y $lib(E)\cap\{x_1,..,x_n\}=\emptyset$:

$\mathfrak{I}(\lambda x_1..x_n.E'[E/x])(a_1,..,a_n) = \mathfrak{I}[\bar a/\bar x](E'[E/x])$ (donde $\bar a$ y $\bar x$ se refieren, de la manera obvia, a las n-tuplas $(a_1,..,a_n)$ y $(x_1,..,x_n)$, respectivamente)

$= \mathfrak{I}[\bar a/\bar x][\mathfrak{I}[\bar a/\bar x](E)/x](E')$ (por hipótesis de inducción)

$= \mathfrak{I}[\bar a/\bar x][\mathfrak{I}(E)/x](E')$ (por coincidencia 3 y ser $lib(E)\cap\{x_1,..,x_n\}=\emptyset$)

$= \mathfrak{I}[\mathfrak{I}(E)/x][\bar a/\bar x](E')$ (pues $x\notin\{x_1,...,x_n\}$)

$= \mathfrak{I}[\mathfrak{I}(E)/x](\lambda x_1..x_n.E')(a_1,..,a_n)$

(ii) $x\in\{x_1,..,x_n\}$ y, sin pérdida de generalidad, supongamos que $lib(E)\cap\{x_1,..,x_n\}=\{x_1\}$:

$\mathfrak{I}((\lambda x_1..x_n.E')[E/x])(a_1,..,a_n) = \mathfrak{I}((\lambda z x_2..x_n.E'[z/x_1][E/x])(a_1,..,a_n)$

$= \mathfrak{I}[a_1/z,\ \bar a/\bar x](E'[z/x_1][E/x])$ (entiéndase que, ahora, $\bar a$ y $\bar x$ se refieren a (n-1)-tuplas)

$= \mathfrak{I}[a_1/z,\ \bar{a}/\bar{x}][\mathfrak{I}(E)/x](E'[z/x_1])$ (por 3, al ser $z \notin lib(E)$ y $lib(E) \cap$
$\cap \{x_2,\ldots,x_n\}=\emptyset$, e hipótesis de inducción)

$= \mathfrak{I}[a_1/z,\ \bar{a}/\bar{x}][\mathfrak{I}(E)/x][a_1/x_1](E')$ (por hipótesis de inducción para E' y
por ser $\mathfrak{I}[a_1/z,\ \bar{a}/\bar{x}][\mathfrak{I}(E)/x](z) = a_1$, pues $z \notin \{x, x_2,\ldots,x_n\}$)

$= \mathfrak{I}[\mathfrak{I}(E)/x][\bar{a}/\bar{x}][a_1/x_1](E')$ (por 3, al ser $x \notin \{x_1,\ldots,x_n\}$ y $z \notin lib(E')$)

$= \mathfrak{I}[\mathfrak{I}(E)/x](\lambda x_1..x_n.E')(a_1,..,a_n)$

(iii) $x \in \{x_1,\ldots,x_n\}$ y supongamos, sin pérdida de generalidad, que $x=x_1$ entonces:

$\mathfrak{I}((\lambda x_1..x_n.E')[E/x]) = \mathfrak{I}(\lambda x_1..x_n.E') = \mathfrak{I}[\mathfrak{I}(E)/x](\lambda x_1..x_n.E')$

por el lema de coincidencia 3, ya que $x=x_1$ no está libre en $\lambda x_1..x_n.E'$. ■

Obsérvese que se mantiene el detalle técnico de exigir $\mathfrak{I}(E) \neq \perp_\sigma$ cuando la variable es de individuo, para impedir dar a dicha variable el valor indefinido.

5.- APROXIMACIONES SINTACTICAS Y APROXIMACIONES SEMANTICAS.

Demostrábamos al final de la sección tercera que el menor punto fijo que denota una expresión funcional $\mu X.M$ de tipo $\tau=\sigma_1..\sigma_n\rightarrow\sigma$ en una interpretación cualquiera $\mathfrak{I}$ era la cota superior mínima de la siguiente cadena de funciones:

$<\mu X.M>^0 = \Omega_\tau$

$<\mu X.M>^{i+1} = \mathfrak{I}[<\mu X.M>^i/X](M)$

Dicha sucesión de funciones corresponde a la forma en que se obtiene el menor punto fijo de la función continua $T(\mathfrak{I}, X, M)$ (cf. 3.4) definida por:

$$T(\mathfrak{I}, X, M)(f) = \mathfrak{I}[f/X](M)$$

para toda $f\in[D^{\sigma_1}\bullet..\bullet D^{\sigma_n}\rightarrow D^{\sigma}]_\bullet$.

De acuerdo con 2.6:

$$\mathfrak{I}(\mu X.M) = fix(T(\mathfrak{I}, X, M)) = \bigcup \{T^i(\mathfrak{I}, X, M)(\Omega_\tau) \ / \ i<\omega\}$$

y, como se vio en 3.5, $T^i(\mathfrak{I}, X, M)(\Omega_\tau) = <\mu X.M>^i$, para todo $i<\omega$.

Por tanto $<\mu X.M>^i \leq \mathfrak{I}(\mu X.M)$, para todo $i<\omega$, y es en este sentido que se dice que $<\mu X.M>^i$ "aproxima", o es una "aproximación" de, la función denotada por $\mathfrak{I}(\mu X.M)$. Como en la definición de $<\mu X.M>^i$ no interviene ninguna expresión del lenguaje, salvo la que se está definiendo, hablamos de *aproximaciones semánticas*, haciendo alusión al hecho de que se obtienen de forma inmediata a partir de la semántica dada.

Cabe pensar si es posible que cada aproximación semántica tenga una contrapartida sintáctica en el lenguaje. Ello es cierto como pasamos a demostrar a continuación.

1.-Lema.

Dadas una Σ-interpretación $\mathfrak{I}$, una variable funcional $X\in VF^\tau$ y una expresión funcional $M\in EF^\tau_\Sigma$, se tiene que las expresiones $[\mu X.M]^i$ definidas por inducción sobre i como:

$[\mu X.M]^0 = \Omega_\tau(\in EF^\tau_\Sigma)$

$[\mu X.M]^{i+1} = M[[\mu X.M]^i/X]$

satisfacen:

(a) $<\mu X.M>^i = [\mu X.M]^i$, para todo $i<\omega$.

(b) $\mathfrak{I}(\mu X.M) = \bigcup \{[\mu X.M]^i \ / \ i<\omega\}$.

Demostración. (a) Por inducción sobre i. Para $i=0$ es evidente y supuesto para i resulta que:

$\mathfrak{I}([\mu X.M]^{i+1}) = \mathfrak{I}(M[[\mu X.M]^i/X]) = \mathfrak{I}[\mathfrak{I}[\mu X.M]^i/X](M)$ (por el lema de sustitución 4.5) =

$= \mathcal{I}[\langle \mu X.M \rangle^i / X](M)$ (por hipótesis de inducción) =

$= \langle \mu X.M \rangle^{i+1}$ (por definición)

(b) es inmediato a partir de (a). ■

Como antes resulta que $\mathcal{I}([\mu X.M]^i) \leq \mathcal{I}(\mu X.M)$, para todo $i<\omega$, y en el mismo sentido del que hablamos, decimos, de nuevo, que $\mathcal{I}([\mu X.M]^i)$ "aproxima" la función que denota $\mathcal{I}(\mu X.M)$. Por razones evidentes podemos entonces llamar, a estas aproximaciones, *sintácticas*.

Ahora bien, si consideramos una expresión funcional (definida sobre una signatura apropiada) tal como $\mu X.\mu Y.\lambda x.(x=0 \rightarrow X(0),\ Y(0))$ y calculamos su segunda aproximación sintáctica resultará la siguiente expresión:

$$\mu Y.\lambda x.(x=0 \rightarrow (\mu Y.\lambda x.x=0 \rightarrow \Omega(0),\ Y(0))(0),\ Y(0))$$

expresión que contiene dos aplicaciones del operador de punto fijo μ. Es fácil ver que, en general, la i-ésima aproximación sintáctica contendrá i veces el operador μ. Pero si nuestro propósito es encontrar una lógica que maneje formalmente el operador μ a través de sus aproximaciones sintácticas y que permita deducir propiedades del primero a partir de propiedades de las segundas, se hace necesario disponer de una medida (de complejidad) que demuestre que cualquier aproximación sintáctica es menos compleja que la expresión que aproxima, y ello se hace extremadamente complicado si tenemos en cuenta que, según muestra el ejemplo anterior, el número de operadores μ lejos de disminuir puede llegar a crecer de forma extraordinaria.

La dificultad proviene de que, siempre que existan operadores μ anidados, esto es, unos en el alcance de otros, la eliminación del más externo produce nuevos anidamientos en los interiores. Para solucionarlo basta con aproximar, a la vez, todas las funciones que aparezcan definidas a través del operador μ. Ello puede hacerse, en principio, de varias formas. La que nosotros seguimos se define como sigue, donde hablamos de aproximaciones sintácticas, calificando las que introdujimos antes como *directas*, ya que entendemos que éstas, al coincidir con las semánticas, reflejan de una manera directa el comportamiento del operador de punto fijo, desde un punto de vista sintáctico.

2.-Definición. *Aproximaciones sintácticas.*

Dadas una Σ-expresión $E \in E_\Sigma^\sigma$, una expresión funcional $M \in EF_\Sigma^\tau$ y una fórmula χ libre de cuantificadores y operador de definición Δ, definimos sus aproximaciones sintácticas, respectivamente, por inducción sobre sus

estructuras, de la siguiente manera:

(1) $E^i \in E^{\sigma}_{\Sigma}$:

$t^i = t$

$(\chi \rightarrow E)^i = \chi^i \rightarrow E^i$

$(\chi \rightarrow E, E')^i = \chi^i \rightarrow E^i, E'^i$

$(M(E_1,\ldots,E_n))^i = M^i(E_1^i,\ldots,E_n^i)$

(2) $M^i \in EF^{\tau}_{\Sigma}$:

$f^i = f$

$X^i = X$

$(\lambda x_1..x_n.E)^i = \lambda x_1..x_n.E^i$

$(\mu X.M)^i$. Se define por inducción sobre i de la forma:

$(\mu X.M)^0 = \Omega_{\tau}$

$(\mu X.M)^{i+1} = M^{i+1}[(\mu X.M)^i/X]$

(3) $\chi \in F'_{\Sigma}$, libre de cuantificadores y operador de definición Δ:

$(E \approx E')^i = E^i \approx E'^i$

$(R(E_1,\ldots,E_n))^i = R(E_1^i,\ldots,E_n^i)$

$(\neg\varphi)^i = \neg\varphi^i$

$(\varphi \vee \psi)^i = \varphi^i \vee \psi^i$ ■

Obsérvese que mientras las aproximaciones sintácticas directas solo se ocupaban de las funciones definidas mediante el operador de punto fijo μ, las aproximaciones sintácticas tratan cualquier expresión (o fórmula, con las restricciones adecuadas). Evidentemente, si una expresión (o fórmula) no contiene ninguna definición recursiva es fácil demostrar que cualquiera de sus aproximaciones sintácticas coincide con ella misma. El ejemplo más claro de esto son los términos.

Para demostrar que las aproximaciones que acabamos de definir realmente "aproximan", será necesario que, por ejemplo, para toda expresión funcional M y toda interpretación J se tenga que $J(M) = \bigcup \{J(M^i) \mid i<\omega\}$. Para ello veamos primeramente que las aproximaciones se comportan monótonamente.

3.-Lema.

Dadas una Σ-expresión $E \in E^{\sigma}_{\Sigma}$, una expresión funcional $M \in EF^{\tau}_{\Sigma}$ y una fórmula χ libre de cuantificadores y operador de definición Δ, se tiene, para toda Σ-interpretación J y todo $i<\omega$:

(a) $\mathcal{I}(E^i) \leq \mathcal{I}(E^{i+1})$

(b) $\mathcal{I}(\chi^i) \leq \mathcal{I}(\chi^{i+1})$

(c) $\mathcal{I}(M^i) \leq \mathcal{I}(M^{i+1})$.

Demostración. Por inducción simultánea sobre la estructura de E, χ y M. Distinguimos pues los siguientes casos:

(a) Para términos es evidente. Para expresiones condicionales $(\chi \rightarrow E,\ E')^i$ distinguimos el valor de χ^i:

(i) $\mathcal{I}(\chi^i) = \perp \Rightarrow \mathcal{I}((\chi \rightarrow E,\ E')^i) = \mathcal{I}(\chi^i \rightarrow E^i,\ E'^i) = \perp_\sigma$ y el resultado es evidente pues $\perp_\sigma$ es elemento mínimo.

(ii) $\mathcal{I}(\chi^i) = \underline{t} \Rightarrow \underline{t} \leq \mathcal{I}(\chi^{i+1}) \Rightarrow \mathcal{I}(\chi^{i+1}) = \underline{t}$ (por ser plano el cpo de los booleanos)

$\Rightarrow \mathcal{I}((\chi \rightarrow E,\ E')^i) = \mathcal{I}(\chi^i \rightarrow E^i,\ E'^i) = \mathcal{I}(E^i) \leq \mathcal{I}(E^{i+1})$ (por hipótesis de inducción)

$= \mathcal{I}(\chi^{i+1} \rightarrow E^{i+1},\ E'^{i+1})$ (pues $\mathcal{I}(\chi^{i+1}) = \underline{t}$) $= \mathcal{I}(\chi \rightarrow E,\ E')^{i+1}$

(iii) $\mathcal{I}(\chi^i) = \underline{f}$ es análogo a (ii).

Análogamente para $(\chi \rightarrow E)^i$.

Por último, para la aplicación de expresiones funcionales tenemos:

$\mathcal{I}((M(E_1,\ldots,E_n))^i) = \mathcal{I}(M^i(E_1^i,\ldots,E_n^i)) = \mathcal{I}(M^i)(\mathcal{I}(E_1^i),\ldots,\mathcal{I}(E_n^i)) \leq$

$\leq \mathcal{I}(M^i)(\mathcal{I}(E_1^{i+1}),\ldots,\mathcal{I}(E_n^{i+1}))$ (por hipótesis de inducción y usando el hecho de que $\mathcal{I}(M^i)$ es una función continua y, por tanto, monótona)

$\leq \mathcal{I}(M^{i+1})(\mathcal{I}(E_1^{i+1}),\ldots,\mathcal{I}(E_n^{i+1}))$ (por hipótesis de inducción)

$= \mathcal{I}(M^{i+1}(E_1^{i+1},\ldots,E_n^{i+1})) = \mathcal{I}((M(E_1,\ldots,E_n))^{i+1})$

(b) Para fórmulas, si su valor booleano es $\perp$, la demostración es inmediata ya que $\perp$ es elemento mínimo. De los otros dos valores booleanos, solo nos ocuparemos de $\underline{t}$, haciendo observar que $\underline{f}$ es dual. Tenemos entonces que:

(i) $\mathcal{I}((E \approx E')^i) = \mathcal{I}(E^i \approx E'^i) = \underline{t} \Rightarrow \mathcal{I}(E^i) = \mathcal{I}(E'^i)$ y ambos son valores definidos, esto es, distintos de $\perp_\sigma$. Ahora bien, por tratarse de cpo's planos y aplicando la hipótesis de inducción resultará entonces que $\mathcal{I}(E^{i+1}) = \mathcal{I}(E^i) = \mathcal{I}(E'^i) = \mathcal{I}(E'^{i+1})$ con lo que $\mathcal{I}(E^{i+1} \approx E'^{i+1}) = \mathcal{I}((E \approx E')^{i+1}) = \underline{t}$.

(ii) $\mathcal{I}((R(E_1,\ldots,E_n)^i)) = \mathcal{I}(R(E_1^i,\ldots,E_n^i)) = \underline{t} \Rightarrow \mathcal{I}(E_j^i) \neq \perp_\sigma$, para todo j=1,..,n (por ser $R^{\mathbb{Z}}$ estricta) $\Rightarrow \mathcal{I}(E_j^i) = \mathcal{I}(E_j^{i+1}) \neq \perp_\sigma$, para todo j=1,..,n (por hipótesis de inducción y por tratarse de cpo's planos) $\Rightarrow$ $\mathcal{I}(R(E_1^i,\ldots,E_n^i)) = \mathcal{I}(R(E_1^{i+1},\ldots,E_n^{i+1})) = \mathcal{I}((R(E_1,\ldots,E_n)^{i+1})) = \underline{t}$.

(iii) Los casos $\neg\varphi$ y $\varphi\vee\psi$ son inmediatos, por hipótesis de inducción.

(c) Para f o X es inmediato. Para expresiones funcionales definidas por abstracción resulta que:

$$\mathcal{I}((\lambda x_1..x_n.E)^i)(a_1,...,a_n) = \mathcal{I}((\lambda x_1..x_n.E^i))(a_1,...,a_n) =$$
$$= \mathcal{I}[a_1/x_1,...,a_n/x_n](E^i) \leq \mathcal{I}[a_1/x_1,...,a_n/x_n](E^{i+1}) \text{ (por hipótesis de inducción)}$$
$$= \mathcal{I}((\lambda x_1..x_n.E^{i+1}))(a_1,...,a_n) = \mathcal{I}((\lambda x_1..x_n.E)^{i+1})(a_1,...,a_n)$$

Para expresiones funcionales definidas recursivamente lo demostramos por inducción sobre i:

i=0. Es inmediato ya que Ω_τ es el elemento mínimo del correspondiente cpo de funciones.

i⟶i+1. Se tiene que:

$$\mathcal{I}((\mu X.M)^{i+1}) = \mathcal{I}(M^{i+1}[(\mu X.M)^i/X]) = \mathcal{I}[\mathcal{I}((\mu X.M)^i)/X](M^{i+1}) \text{ (por el lema de sustitución)}$$
$$\leq \mathcal{I}[\mathcal{I}((\mu X.M)^i)/X](M^{i+2}) \text{ (por hip. de inducción sobre } M^{i+1})$$
$$\leq \mathcal{I}[\mathcal{I}((\mu X.M)^{i+1})/X](M^{i+2}) \text{ (por hipótesis de inducción sobre i y monotonía de la semántica de } M^{i+2})$$
$$= \mathcal{I}(M^{i+2}[(\mu X.M)^{i+1}/X]) \text{ (por el lema de sustitución 4.5)}$$
$$= \mathcal{I}((\mu X.M)^{i+2})$$

como queríamos demostrar. ■

Demostrada la monotonía de las aproximaciones sintácticas veamos que la cota superior mínima de la interpretación de cada una de ellas, coincide con la denotación de la expresión que están aproximando.

4.-Teorema. *Equivalencia de aproximaciones sintácticas y semánticas.*

Dadas una Σ-expresión $E\in E^{\sigma}_{\Sigma}$, una expresión funcional $M\in EF^{\tau}_{\Sigma}$ y una fórmula χ libre de cuantificadores y operador de definición Δ, se tiene, para toda Σ-interpretación $\mathcal{I}$:

(a) $\mathcal{I}(E) = \bigcup\{\mathcal{I}(E^i) \mid i<\omega\}$

(b) $\mathcal{I}(\chi) = \bigcup\{\mathcal{I}(\chi^i) \mid i<\omega\}$

(c) $\mathcal{I}(M) = \bigcup\{\mathcal{I}(M^i) \mid i<\omega\}$.

Demostración. Por inducción simultánea sobre la estructura de E, χ y M. Se tienen pues los siguientes casos:

(a) Para términos es evidente. Para expresiones condicionales $\chi \rightarrow E, E'$ distinguimos según sea el valor de χ:

(i) $\mathcal{I}(\chi) = \perp \Leftrightarrow \mathcal{I}(\chi^i) = \perp$, para todo $i<\omega$ (por hipótesis de inducción)

Luego $\mathcal{I}(\chi \rightarrow E, E') = \bigsqcup \{\mathcal{I}((\chi \rightarrow E, E')^i) / i<\omega\} = \perp_\sigma$ (como acabamos de ver).

(ii) $\mathcal{I}(\chi) = \underline{t} \Leftrightarrow$ existe $i<\omega$ tal que $\mathcal{I}(\chi^i) = \underline{t}$ (por hip. ind.)

Luego, considerando dicho i, resulta que:

$\mathcal{I}(\chi \rightarrow E, E') = \mathcal{I}(E) = \bigsqcup \{\mathcal{I}(E^j) / j<\omega\}$ (por hip. de ind.) =

$= \bigsqcup \{\mathcal{I}((\chi \rightarrow E, E')^j) / j \geq i\}$ (pues si $\mathcal{I}(\chi^i) = \underline{t}$ entonces, por monotonía, tendremos que $\mathcal{I}(\chi^j) = \underline{t}$, para $j \geq i$) =

$= \bigsqcup \{\mathcal{I}((\chi \rightarrow E, E')^j) / j<\omega\}$ (pues si $\mathcal{I}(\chi^i) = \underline{t}$ entonces, por monotonía, tendremos que $\mathcal{I}(\chi^j) \in \{\perp, \underline{t}\}$ con lo que, de nuevo por monotonía, esta vez referida a E, el supremo sigue siendo el mismo)

Análogamente se demuestra cuando $\mathcal{I}(\chi) = \underline{f}$. De forma similar para $\chi \rightarrow E$.

Para el caso de expresiones obtenidas por aplicación se tiene:

$\mathcal{I}(M(E_1,\ldots,E_n)) = \mathcal{I}(M)(\mathcal{I}(E_1),\ldots,\mathcal{I}(E_n)) =$

$= [\bigsqcup \{\mathcal{I}(M^i) / i<\omega\}]([\bigsqcup \{\mathcal{I}(E_1^i) / i<\omega\}],\ldots,[\bigsqcup \{\mathcal{I}(E_n^i) / i<\omega\}])$ (hip. ind.)

$= [\bigsqcup \{\mathcal{I}(M^i) / i<\omega\}][\bigsqcup \{(\mathcal{I}(E_1^i),\ldots,\mathcal{I}(E_n^i)) / i<\omega\}]$ (pues la operación estricta de construcción de tuplas es inmediato demostrar que es continua)

$= \bigsqcup \{\mathcal{I}(M^i)[\bigsqcup \{(\mathcal{I}(E_1^i),\ldots,\mathcal{I}(E_n^i)) / i<\omega\}] / i<\omega\}$ (por definición de $\bigsqcup \{f_i / i<\omega\}$)

$= \bigsqcup \{\bigsqcup \{\mathcal{I}(M^i)(\mathcal{I}(E_1^j),\ldots,\mathcal{I}(E_n^j)) / j<\omega\} / i<\omega\}$ (por continuidad de M^i)

$= \bigsqcup \{\mathcal{I}(M^i)(\mathcal{I}(E_1^i),\ldots,\mathcal{I}(E_n^i)) / i<\omega\}$ (la relación $\geq$ es evidente; para $\leq$ úsese la monotonía que acabamos de demostrar)

$= \bigsqcup \{\mathcal{I}((M(E_1,\ldots,E_n)^i)) / i<\omega\}$

(b) Respecto a las fórmulas, para la igualdad se tienen los casos:

(i) $\mathcal{I}(E \approx E') = \perp$. Entonces $\mathcal{I}(E) = \perp_\sigma$ o $\mathcal{I}(E') = \perp_\sigma$. Supongamos, sin pérdida de generalidad, que $\mathcal{I}(E) = \perp_\sigma$. Por hipótesis de inducción tendremos que $\bigsqcup \{\mathcal{I}(E^i) / i<\omega\} = \perp_\sigma$ con lo que $\mathcal{I}(E^i) = \perp_\sigma$, para todo $i<\omega$. Por tanto se tendrá $\bigsqcup \{\mathcal{I}(E^i \approx E'^i) / i<\omega\} = \perp$, pues la igualdad se interpreta como una función estricta. Luego $\bigsqcup \{\mathcal{I}((E \approx E')^i) / i<\omega\} = \perp$.

Recíprocamente si $\bigcup \{\mathcal{I}((E \approx E')^i) \mid i<\omega\} = \perp$ entonces $\mathcal{I}(E^i) = \perp_\sigma$ o $\mathcal{I}(E'^i) = \perp_\sigma$, para todo $i<\omega$. Si $\mathcal{I}(E^i) = \perp_\sigma$ (análogamente para E'^i), para todo $i<\omega$, entonces es evidente que $\mathcal{I}(E \approx E') = \perp$, pues aplicando la hipótesis de inducción tendríamos que $\mathcal{I}(E) = \bigcup \{\mathcal{I}(E^i) \mid i<\omega\} = \perp_\sigma$. Si, por el contrario, existe $i<\omega$ tal que $\mathcal{I}(E^i) \neq \perp_\sigma$ entonces $\mathcal{I}(E^j) \neq \perp_\sigma$, para todo $j \geq i$, con lo que $\mathcal{I}(E'^j) = \perp_\sigma$, para todo $j \geq i$ y, por hipótesis de inducción, $\bigcup \{\mathcal{I}(E'^i) \mid i<\omega\} = \mathcal{I}(E') = \perp_\sigma$. Por tanto, en ambos casos, uno de los miembros de la igualdad toma valor indefinido y, en consecuencia, la igualdad también.

(ii) $\mathcal{I}(E \approx E') = \underline{t}$. Entonces $\mathcal{I}(E) = \mathcal{I}(E') \neq \perp_\sigma$. Por hipótesis de inducción $\bigcup \{\mathcal{I}(E^i) \mid i<\omega\} = \bigcup \{\mathcal{I}(E'^i) \mid i<\omega\} \neq \perp_\sigma$. Por tratarse de cpo's planos, existen $i, j<\omega$ tales que $\mathcal{I}(E^i) = \mathcal{I}(E'^j) \neq \perp_\sigma$ y, para todo $k>\max\{i, j\}$ se tiene $\mathcal{I}(E^k) = \mathcal{I}(E'^k) \neq \perp_\sigma$. Por tanto tendremos que: $\bigcup \{\mathcal{I}(E^i \approx E'^i) \mid i<\omega\} = \bigcup \{\mathcal{I}((E \approx E')^i) \mid i<\omega\} = \underline{t}$. El recíproco sigue una demostración similar; téngase en cuenta para ello que si $\mathcal{I}(E^i) \neq \perp_\sigma$, para algún $i<\omega$, entonces $\mathcal{I}(E) = \mathcal{I}(E^i)$. Por último decir que el caso en que la igualdad es falsa se demuestra análogamente.

El caso de los predicados es parecido y la negación y disyunción son inmediatas, por hipótesis de inducción.

(c) La demostración para f o X es trivial. Para el caso de la abstracción tenemos:

$$(\bigcup \{\mathcal{I}(\lambda x_1..x_n.E^i) \mid i<\omega\})(a_1,...,a_n) = \bigcup \{\mathcal{I}(\lambda x_1..x_n.E^i)(a_1,...,a_n) \mid i<\omega\} =$$

$$= \bigcup \{\mathcal{I}[a_1/x_1,...,a_n/x_n](E^i) \mid i<\omega\} \text{ (por definición)}$$

$$= \mathcal{I}[a_1/x_1,...,a_n/x_n](E) \text{ (por hipótesis de inducción)}$$

$$= \mathcal{I}(\lambda x_1..x_n.E)(a_1,...,a_n)$$

Por último, para el caso de la recursión demostramos los dos sentidos del orden $\underline{\leq}$. Se tiene:

(i) $\underline{\geq}$. Por inducción sobre i:

i=0. Es evidente pues Ω_τ es elemento mínimo.

i⟶i+1. Resulta que:

$$\mathcal{I}((\mu X.M)^{i+1}) = \mathcal{I}(M^{i+1}[(\mu X.M)^i/X]) = \mathcal{I}[\mathcal{I}(\mu X.M)^i/X](M^{i+1}) \text{ (por el lema de sustitución 4.5)}$$

$$\underline{\leq} \mathcal{I}[\mathcal{I}(\mu X.M)^i/X](M) \text{ (por hipótesis de inducción para M)}$$

$$\underline{\leq} \mathcal{I}[\mathcal{I}(\mu X.M)/X](M) \text{ (por hipótesis de inducción para i y monotonía de la semántica de M (cf. 3.4.(c)))}$$

$$= \mathcal{I}(\mu X.M) \text{ (por definición de la semántica de } \mu X.M)$$

(ii) $\leq$. Demostremos que $\langle \mu X.M \rangle^i \leq \bigcup \{\mathcal{I}((\mu X.M)^j) \mid j<\omega\}$, para todo $i<\omega$, y entonces $\bigcup \{\langle \mu X.M \rangle^i \mid i<\omega\} = \mathcal{I}(\mu X.M) \leq \bigcup \{\mathcal{I}((\mu X.M)^j) \mid j<\omega\}$, como se busca. Demostramos esto por inducción sobre i:

i=0. Inmediato como antes.

$i \rightarrow i+1$. Se tiene que:

$\langle \mu X.M \rangle^{i+1} = \mathcal{I}[\langle \mu X.M \rangle^i / X](M) \leq \mathcal{I}[\bigcup \{\mathcal{I}((\mu X.M)^j) \mid j<\omega\}/X](M)$ (por hipótesis de inducción sobre i y monotonía de la semántica de M (cf. 3.4.(c)))

$= \bigcup \{\mathcal{I}[\mathcal{I}((\mu X.M)^j)/X](M) \mid j<\omega\}$ (por continuidad de T; cf. 3.4.(c))

$\leq \bigcup \{\bigcup \{\mathcal{I}[\mathcal{I}((\mu X.M)^j)/X](M^i) \mid i<\omega\} \mid j<\omega\}$ (por hip. ind. sobre M)

$\leq \bigcup \{\mathcal{I}[\mathcal{I}((\mu X.M)^k)/X](M^k) \mid k<\omega\}$ (dados i, j<ω, tomar k=max{i, j} y aplicar monotonía de M^i)

$\leq \bigcup \{\mathcal{I}[\mathcal{I}((\mu X.M)^k)/X](M^{k+1}) \mid k<\omega\}$ (por monotonía de M^k)

$= \bigcup \{\mathcal{I}(M^{k+1}[(\mu X.M)^k/X]) \mid k<\omega\}$ (por el lema de sustitución 4.5)

$= \bigcup \{\mathcal{I}((\mu X.M)^{k+1}) \mid k<\omega\}$ (por definición)

Como la relación $\leq$ es antisimétrica, de (i) y (ii) resulta lo que queríamos demostrar. ∎

6.- UN METODO DE TABLEAUX PARA LFRP. PRELIMINARES.

El sentido del método de los tableaux dentro de la lógica y su forma de operar fueron explicados ya en el capítulo anterior cuando se introdujeron para LFP. Por ello, nos limitaremos aquí a explicar qué modificaciones son necesarias a fin de que puedan aplicarse también a la lógica LFRP.

Como se hizo allí, extenderemos la signatura (heterogénea) Σ, con una colección de conjuntos numerables de nuevas constantes, que llamaremos auxiliares, $C^{\sigma} = \{c_n^{\sigma} \ / \ n<\omega\}$, para cada género $\sigma \in S$. Siempre que no haya ambigüedad omitiremos el tipo. A la signatura así extendida la representaremos mediante $\overline{\Sigma}$.

Como estos nuevos símbolos denotarán individuos definidos con ciertas propiedades, nos restringiremos, como hicimos en el capítulo anterior, a aquellas $\overline{\Sigma}$-interpretaciones $\overline{\mathfrak{I}}$ tales que $\overline{\mathfrak{I}}(c^{\sigma}) \neq \perp_{\sigma}$, para todo $c^{\sigma} \in C^{\sigma}$ y todo $\sigma \in S$.

Para la construcción de tableaux habremos de tener en cuenta los nuevos tipos de expresiones, haciendo notar que algunos de ellos darán lugar a la bifurcación de un nodo en una cantidad numerable de ramas.

1.-<u>Definición</u>. *Clases BASICA, ALFA, BETA, GAMMA, DELTA, ω-ALFA, ω-BETA.*

Una *clasificación orientada hacia los tableaux* (de LFRP) es una partición del conjunto de las $\overline{\Sigma}$-fórmulas en siete clases disjuntas de manera que se satisfagan los siguientes requerimientos:

(1)-(3) Los requerimientos para las clases BASICA, ALFA y BETA son los mismos que los que se dieron en §2.3.1.

(4) GAMMA: Está formada por el conjunto de las llamadas <u>fórmulas universales</u> γ que deben satisfacer:

$$\mathrm{Sat}_{\Sigma} \Phi \cup \{\gamma\} \Leftrightarrow \mathrm{Sat}_{\Sigma} \Phi \cup \{\gamma, \gamma(E)\}$$

para toda $\overline{\Sigma}$-expresión $E \in E_{\overline{\Sigma}}^{\sigma}$, siendo γ de la forma $\neg\exists x^{\sigma}\varphi$.

Su esquema es:

$$\frac{\gamma}{\gamma(E)} \ (E \in E_{\overline{\Sigma}}^{\sigma})$$

(5) DELTA: Está formada por el conjunto de las llamadas <u>fórmulas existenciales</u> δ que deben satisfacer:

$$\mathrm{Sat}_{\Sigma} \Phi \cup \{\delta\} \Leftrightarrow \mathrm{Sat}_{\Sigma} \Phi \cup \{\delta(c^{\sigma})\}$$

para toda constante auxiliar $c^{\sigma} \in C^{\sigma}$, siendo δ de la forma $\exists x^{\sigma}\varphi$, que no aparezca ni en Φ ni en δ. El esquema es como el que se dio en §2.3.1.

(6) ω-ALFA: Está formada por el conjunto de las llamadas fórmulas ω-conjuntivas $\alpha^{\bullet}$ que deben satisfacer:

$Sat_{\mathfrak{A}} \Phi \cup \{\alpha^{\bullet}\} \Leftrightarrow Sat_{\mathfrak{A}} \Phi \cup \{\alpha^{\bullet}_{i} / i<\omega\}$

para cierta cantidad numerable de constituyentes $\alpha^{\bullet}_{i}$, $i<\omega$.

Su esquema es:

$$\frac{\alpha^{\bullet}}{\alpha^{\bullet}_{i}} \ (i<\omega)$$

(7) ω-BETA: Está formada por el conjunto de las llamadas fórmulas ω-disyuntivas $\beta^{\bullet}$ que deben satisfacer:

$Sat_{\mathfrak{A}} \Phi \cup \{\beta^{\bullet}\} \Leftrightarrow$ existe $i<\omega$ tal que $Sat_{\mathfrak{A}} \Phi \cup \{\beta^{\bullet}_{i}\}$

para cierta cantidad numerable de constituyentes $\beta^{\bullet}_{i}$, $i<\omega$.

Su esquema es:

$$\frac{\beta^{\bullet}}{\beta^{\bullet}_{i} \ (i<\omega)}$$

En todos los apartados anteriores, Φ representa un conjunto arbitrario de $\bar{\Sigma}$-fórmulas. ■

Obsérvese que las fórmulas universales y existenciales se comportan como las dadas para LFP, aunque se generaliza al conjunto de expresiones de tipo elemental. Asimismo, las fórmulas ω-conjuntivas y ω-disyuntivas extienden el caso de una conjunción o una disyunción a una cantidad numerable de miembros y su comportamiento es, en este sentido, análogo.

Para nuestra lógica LFRP, las condiciones (1)-(7) de la definición anterior dan lugar a la siguiente partición:

BASICA: Δx, $\neg\Delta x$, para cualquier variable de individuo de cualquier género σ, Δc, $\neg\Delta c$, para cualquier constante c de $\bar{\Sigma}$, $\neg\Delta f(E_1,\ldots,E_n)$, $\neg\Delta R(E_1,\ldots,E_n)$, $\neg\Delta X(E_1,\ldots,E_n)$,

ALFA:

$$(\alpha 1)\ \frac{R(E_1,\ldots,E_n)}{\begin{array}{c}\Delta E_1\\ \vdots\\ \Delta E_n\end{array}} \qquad (\alpha 2)\ \frac{\neg R(E_1,\ldots,E_n)}{\begin{array}{c}\Delta E_1\\ \vdots\\ \Delta E_n\end{array}} \qquad (\alpha 3)\ \frac{E_1 \approx E_2}{\begin{array}{c}\Delta E_1\\ \Delta E_2\end{array}}$$

$$(\alpha 4)\ \frac{\neg E_1 \propto E_2}{\Delta E_1 \quad \Delta E_2} \qquad (\alpha 5)\ \frac{\neg\neg\varphi}{\varphi} \qquad (\alpha 6)\ \frac{\neg(\varphi \vee \psi)}{\neg\varphi \quad \neg\psi}$$

$$(\alpha 7)\ \frac{\Delta f(E_1,\ldots,E_n)}{\Delta E_1 \quad \vdots \quad \Delta E_n} \qquad (\alpha 8)\ \frac{\Delta X(E_1,\ldots,E_n)}{\Delta E_1 \quad \vdots \quad \Delta E_n}$$

$$(\alpha 9)\ \frac{\Delta(\lambda x_1 \ldots x_n . E)(E_1,\ldots,E_n)}{\Delta E[E_1/x_1,\ldots,E_n/x_n] \quad \Delta E_i \quad \vdots \quad \Delta E_n} \qquad (\alpha 10)\ \frac{\Delta(\chi \rightarrow E)}{\chi \quad \Delta E}$$

BETA:

$$(\beta 1)\ \frac{\varphi \vee \psi}{\varphi \mid \psi} \qquad (\beta 2)\ \frac{\Delta(\chi \rightarrow E,\ E')}{\chi \wedge \Delta E \mid \neg\chi \wedge \Delta E'}$$

$$(\beta 3)\ \frac{\neg\Delta(\chi \rightarrow E,\ E')}{\chi \wedge \neg\Delta E \mid \neg\chi \wedge \neg\Delta E' \mid \neg\Delta\chi} \qquad (\beta 4)\ \frac{\Delta R(E_1,\ldots,E_n)}{R(E_1,\ldots,E_n) \mid \neg R(E_1,\ldots,E_n)}$$

$$(\beta 5)\ \frac{\neg\Delta(\chi \rightarrow E)}{\chi \wedge \neg\Delta E \mid \neg\chi \mid \neg\Delta\chi}$$

$$(\beta 6)\ \frac{\neg\Delta(\lambda x_1 \ldots x_n . E)(E_1,\ldots,E_n)}{\neg\Delta E[E_1/x_1,\ldots,E_n/x_n] \mid \neg\Delta E_1 \mid \ldots \mid \neg\Delta E_n}$$

GAMMA:

$$(\gamma)\ \frac{\neg\exists x\varphi}{\Delta E \rightarrow \neg\varphi[E/x]} \quad (x \in V^{\sigma},\ E \in E^{\sigma}_{\Sigma})$$

DELTA:

$$(\delta)\ \frac{\exists x\varphi}{\varphi[c/x]}\quad (x\in V^{\sigma},\ c\in C^{\sigma})$$

ω-ALFA:

$$(\alpha^{*})\ \frac{\neg\Delta(\mu X.M)(E_1,\ldots,E_n)}{\neg\Delta(\mu X.M)^{i}(E_1,\ldots,E_n)}\quad (i<\omega)$$

ω-BETA:

$$(\beta^{*})\ \frac{\Delta(\mu X.M)(E_1,\ldots,E_n)}{\Delta(\mu X.M)^{i}(E_1,\ldots,E_n)\quad (i<\omega)}$$

Para comprobar que (1)-(7) de la definición anterior dan una partición de las fórmulas basta seguir el siguiente esquema donde a la derecha de cada tipo de fórmula aparece la condición de la clasificación anterior en la que se incluye:

$(E_1 \approx E_2)$				$(\alpha 3)$
$R(E_1,\ldots,E_n)$				$(\alpha 1)$
ΔE:	Δt:	Δc		BASICA
		Δx		BASICA
		$\Delta f(t_1,\ldots,t_n)$		$(\alpha 7)$
	$\Delta(\chi\rightarrow E)$			$(\alpha 10)$
	$\Delta(\chi\rightarrow E,\ E')$			$(\beta 2)$
	$\Delta M(E_1,\ldots,E_n)$:			
		$\Delta f(E_1,\ldots,E_n)$		$(\alpha 7)$
		$\Delta X(E_1,\ldots,E_n)$		$(\alpha 8)$
		$\Delta(\lambda x_1..x_n.E)(E_1,\ldots,E_n)$		$(\alpha 9)$
		$\Delta(\mu X.M)(E_1,\ldots,E_n)$		(β^{*})
$\Delta R(E_1,\ldots,E_n)$				$(\beta 4)$
$\neg\varphi$:	$\neg(E_1 \approx E_2)$			$(\alpha 4)$
	$\neg R(E_1,\ldots,E_n)$			$(\alpha 2)$
	$\neg\Delta E$:	$\neg\Delta t$:	$\neg\Delta c$	BASICA
			$\neg\Delta x$	BASICA
			$\neg\Delta f(t_1,\ldots,t_n)$	BASICA
		$\neg\Delta(\chi\rightarrow E)$		$(\beta 5)$
		$\neg\Delta(\chi\rightarrow E,\ E')$		$(\beta 3)$
		$\neg\Delta M(E_1,\ldots,E_n)$:		
			$\neg\Delta f(E_1,\ldots,E_n)$	BASICA
			$\neg\Delta X(E_1,\ldots,E_n)$	BASICA

	$\neg\Delta(\lambda x_1..x_n.E)(E_1,...,E_n)$	$(\beta 6)$
	$\neg\Delta(\mu X.M)(E_1,...,E_n)$	(α^*)
$\neg\Delta R(E_1,...,E_n)$		BASICA
$\neg\neg\varphi_1$		$(\alpha 5)$
$\neg(\varphi_1 \vee \varphi_2)$		$(\alpha 6)$
$\neg\exists x\varphi_1$		(γ)
$\varphi_1 \vee \varphi_2$		$(\beta 1)$
$\exists x\varphi$		(δ)

2.-Lema.

La clasificación que acabamos de dar es una clasificación orientada hacia los tableaux, en el sentido de la definición anterior.

Demostración. Como en §2.3.2, veamos que se satisfacen los requerimientos semánticos (2)-(7) que aparecían en la definición anterior. Para ello demostramos algo aún más fuerte, a saber: dada una $\bar{\Sigma}$-interpretación $\mathfrak{I}$ cualquiera, se tiene:

ALFA: $\mathfrak{I} \models_s \alpha \Rightarrow \mathfrak{I} \models_s \alpha_1 \wedge .. \wedge \alpha_n$

Además "$\Leftarrow$" vale también excepto en los casos $(\alpha 1)$, $(\alpha 2)$, $(\alpha 3)$, $(\alpha 4)$, $(\alpha 7)$ y $(\alpha 8)$.

BETA: $\mathfrak{I} \models_s \beta \Leftrightarrow \mathfrak{I} \models_s \beta_1 \vee .. \vee \beta_n$

GAMMA: $\mathfrak{I} \models_s \gamma \Rightarrow \mathfrak{I} \models_s \gamma(E)$, para cualquier $\bar{\Sigma}$-expresión E

DELTA: $\mathfrak{I} \models_s \Phi \cup \{\delta\} \Leftrightarrow \mathfrak{I}[a/c] \models_s \Phi \cup \{\delta(c)\}$, para algún elemento a del dominio de $\mathfrak{I}$, siempre que $c \in C$ sea nueva, esto es, no aparezca ni en Φ ni en δ

ω-ALFA: $\mathfrak{I} \models_s \alpha^* \Leftrightarrow \mathfrak{I} \models_s \alpha_i^*$, para todo $i<\omega$

ω-BETA: $\mathfrak{I} \models_s \beta^* \Leftrightarrow \mathfrak{I} \models_s \beta_i^*$, para algún $i<\omega$.

La demostración de $(\alpha 1)$-$(\alpha 7)$ es similar, cuando no la misma, a la dada en el capítulo anterior (cf. lema citado). Para $(\alpha 8)$ cabe decir algo parecido. Obsérvese que si $\Delta X(E_1,...,E_n)$ es cierta entonces $\mathfrak{I}(X(E_1,...,E_n)) = \varepsilon(X)(\mathfrak{I}(E_1),...,\mathfrak{I}(E_n))$ está definido con lo que, por ser $\varepsilon(X)$ una función estricta, supone que $\mathfrak{I}(E_i)$ esté definido, para todo $i=1,..,n$.

Para $(\alpha 9)$ se tiene, por hipótesis:

$$\mathfrak{I}((\lambda x_1..x_n.E)(E_1,...,E_n)) = \mathfrak{I}((\lambda x_1..x_n.E))(\mathfrak{I}(E_1),..,\mathfrak{I}(E_n)) \neq \perp_\sigma$$

Como los constituyentes en $(\alpha 9)$ solo pueden tener valor booleano cierto o falso, lo mismo le ocurre a su conjunción. Supongamos que $\mathfrak{I}(\Delta E_i) = \underline{f}$, para algún $i=1,..,n$. Entonces $\mathfrak{I}(E_i) = \perp_{\sigma i}$ luego, $(\mathfrak{I}(E_1),...,\mathfrak{I}(E_n)) = \perp_\varnothing$ y, por tratarse de funciones estrictas, tendríamos:

$$\mathfrak{I}((\lambda x_1..x_n.E))(\mathfrak{I}(E_1),...,\mathfrak{I}(E_n)) = \perp_\sigma$$

contra lo supuesto. Ahora bien, si $\mathfrak{I}(\Delta E_i) = \underline{t}$, para todo i=1,..,n, entonces $\mathfrak{I}(E_i) \neq \perp_{\sigma_i}$, para todo i=1,..,n, con lo que tendremos que:

$$\mathfrak{I}((\lambda x_1..x_n.E))(\mathfrak{I}(E_1),...,\mathfrak{I}(E_n)) = \mathfrak{I}[\mathfrak{I}(E_1)/x_1,...,\mathfrak{I}(E_n)/x_n](E) =$$
$$= \mathfrak{I}(E[E_1/x_1,...,E_n/x_n]) \text{ (por el lema de sustitución)}$$

y en consecuencia $\mathfrak{I}(\Delta E[E_1/x_1,...,E_n/x_n]) = \underline{t}$, como buscábamos.

La demostración del recíproco es inmediata ya que si los constituyentes son ciertos entonces la última serie de igualdades que hemos escrito se verifica. Por último, la demostración para (α10) se deduce de la dada en §2.3.2 para (α8).

Las demostraciones de (β1), (β2), (β3), (β4) y (β5) sos análogas a las de (β1), (β2), (β3), (β7) y (β4) vistas en el citado lema §2.3.2. Lo mismo se puede decir sobre la demostración de (γ) y (δ). La demostración de (β6) se sigue de forma inmediata de lo hecho en (α9).

Para (α^*) y (β^*) tenemos:

$$\mathfrak{I}(\Delta(\mu X.M)(E_1,..,E_n)) = \underline{t} \Leftrightarrow \mathfrak{I}((\mu X.M)(E_1,...,E_n)) \neq \perp_\sigma \Leftrightarrow$$
$$\Leftrightarrow \bigcup_i \mathfrak{I}((\mu X.M)^i(E_1,..,E_n)) \neq \perp_\sigma \text{ (por el teorema 5.4)} \Leftrightarrow$$
$$\Leftrightarrow \text{existe } i<\omega \text{ tal que } \mathfrak{I}((\mu X.M)^i(E_1,..,E_n)) \neq \perp_\sigma \text{ (por ser } D^\sigma \text{ cpo plano)}$$
$$\Leftrightarrow \text{existe } i<\omega \text{ tal que } \mathfrak{I}(\Delta(\mu X.M)^i(E_1,..,E_n)) = \underline{t}$$

y la demostración es ahora inmediata. ■

7.- TABLEAUX PARA LFRP.

La definición de los tableaux para la nueva lógica habrá de tener en cuenta la construcción de tableaux ω-ramificados, esto es, árboles que pueden tener nodos con una cantidad numerable de hijos. A diferencia de los que se definieron para LFP, sin embargo, ahora se tratará de árboles con conjuntos de fórmulas etiquetando sus nodos. La razón de hacer esto así quedará patente más adelante. En primer lugar definimos el concepto de conjunto coherente, que se ajusta al que se dió en el capítulo anterior.

1.-<u>Definición</u>. *Conjuntos coherentes. Ramas cerradas y abiertas.*

Un conjunto de $\bar{\Sigma}$-fórmulas Φ se dice coherente si satisface las siguientes condiciones:

(1) No existe ninguna fórmula φ de manera que φ, $\neg\varphi\in\Phi$ o φ, $\neg\Delta\varphi\in\Phi$ o $\neg\varphi$, $\neg\Delta\varphi\in\Phi$.

(2) $\neg\Delta x \notin \Phi$, para ninguna variable $x\in V^{\sigma}$, de cualquier género σ

(3) $\neg\Delta c \notin \Phi$, para ninguna constante auxiliar $c\in C^{\sigma}$, de cualquier género σ.

Dado un árbol $\mathcal{I}$ con conjuntos de $\overline{\Sigma}$-fórmulas del lenguaje LFRP etiquetando sus nodos, decimos que una rama de $\mathcal{I}$ está cerrada si la unión de los conjuntos de fórmulas que etiquetan sus nodos no es coherente. Una rama que no está cerrada diremos que está abierta. ∎

En estas condiciones podemos definir los tableaux, distinguiendo entre los finitos y las sucesiones de estos últimos, teniendo en cuenta que los primeros se entienden finitos solamente en "profundidad", ya que pueden tener una cantidad numerable de ramas. La profundidad de un árbol $\mathcal{I}$ tal que todas sus ramas contienen un número finito de nodos (cf. [Rog 67]) se define por inducción sobre su estructura de la siguiente manera:

(i) si $\mathcal{I}$ está formado por un solo nodo entonces $d(\mathcal{I}) = 0$

(ii) si $\mathcal{I}$ está formado por la raíz y los hijos $\mathcal{I}_i$, $i\in I$, entonces $d(\mathcal{I})$ es el supremo de los ordinales $d(\mathcal{I}_i)+1$, $i\in I$.

Entonces un árbol $\mathcal{I}$ se dirá de profundidad finita si $d(\mathcal{I})$ es un ordinal finito.

La definición de un tableau tal que todas sus ramas contienen un número finito de nodos es, entonces, la siguiente.

2.-Definición. *Extensiones de tableaux y tableaux de ramas finitas.*

Sea $\mathcal{I}$ un árbol ω-ramificado con nodos etiquetados por conjuntos de $\overline{\Sigma}$-fórmulas y sea R una rama finita abierta de $\mathcal{I}$. El árbol $\mathcal{I}'$ que resulta de analizar la etiqueta Ψ de la hoja de R se define por el siguiente proceso:

(A) Si una fórmula conjuntiva α pertenece a Ψ y α_i, para cierto $i\in\{1,..,m\}$, es un constituyente de α que no pertenece a Ψ, entonces ampliar la rama R con un nuevo nodo etiquetado con $\Psi \cup \{\alpha_i\}$.

(B) Si una fórmula disyuntiva β pertenece a Ψ entonces bifurcar la rama R con m nuevos nodos etiquetados con $\Psi \cup \{\beta_i\}$, $i=1,..,m$.

(C) Decimos que una $\overline{\Sigma}$-expresión E de género σ es *adecuada* a un conjunto de fórmulas Ψ si y solo si E se puede construir con símbolos de $\overline{\Sigma}$ y variables que aparecen libres en fórmulas de Ψ, o E es la primera constante auxiliar $c_0^{\sigma}\in C^{\sigma}$.

Si una fórmula universal γ pertenece a Ψ y E es una $\overline{\Sigma}$-expresión adecuada a Ψ, entonces ampliar la rama R con un nuevo nodo etiquetado con el conjunto $\Psi \cup \{\gamma(E)\}$.

(D) Si una fórmula existencial δ pertenece a Ψ y c es una constante auxiliar que no aparece en Ψ, entonces ampliar la rama R con un nuevo nodo etiquetado con $\Psi \cup \{\delta(c)\}$.

(A^*) Si una fórmula ω-conjuntiva α^* pertenece a Ψ y α^*_i, para cierto $i<\omega$, es un constituyente de α^* que no pertenece a Ψ, entonces ampliar la rama R con un nuevo nodo etiquetado con $\Psi \cup \{\alpha^*_i\}$.

(B^*) Si una fórmula ω-disyuntiva β^* pertenece a Ψ, entonces bifurcar la rama R con ω nuevos nodos etiquetados con $\Psi \cup \{\beta^*_i\}$, $i<\omega$.

Representaremos mediante $\Sigma \xrightarrow[R]{} \Sigma'$ el paso de Σ a Σ' a través del análisis, que acabamos de describir, de la etiqueta de la hoja de la rama R.

Dado el árbol ω-ramificado Σ, sea B un conjunto de ramas abiertas de Σ. Escribimos $\Sigma \xrightarrow[B]{} \Sigma'$ para representar que Σ' es el árbol resultante de analizar simultáneamente todas las ramas $R \in B$ de Σ.

Llamamos *axioma de la igualdad* θ a cualquier fórmula de alguno de los siguientes tipos:

$\forall x(x \approx x)$

$\forall x \forall y(x \approx y \rightarrow y \approx x)$

$\forall x \forall y \forall z(x \approx y \wedge y \approx z \rightarrow x \approx z)$

$\forall \bar{x} \forall \bar{y}(\bar{x} \approx \bar{y} \wedge \Delta f(\bar{x}) \rightarrow \Delta f(\bar{y}))$

$\forall \bar{x} \forall \bar{y}(\bar{x} \approx \bar{y} \wedge \Delta X(\bar{x}) \rightarrow \Delta X(\bar{y}))$

$\forall \bar{x} \forall \bar{y}(\bar{x} \approx \bar{y} \wedge \Delta \chi(\bar{x}) \wedge \chi(\bar{x}) \rightarrow \chi(\bar{y}))$

$\forall x(\chi \wedge \Delta \chi \rightarrow (\chi \rightarrow x) \approx x)$

$(\chi \wedge \Delta \chi \wedge \Delta E \rightarrow (\chi \rightarrow E,\ E') \approx E) \wedge$

$\wedge\ (\neg \chi \wedge \Delta \chi \wedge \Delta E' \rightarrow (\chi \rightarrow E,\ E') \approx E')$

donde $E,\ E' \in E^{\sigma}_{\overline{\Sigma}}$

$\forall y_1 .. \forall y_n (\Delta E[y_1/x_1, \ldots, y_n/x_n] \rightarrow$

$\rightarrow (\lambda x_1 .. x_n . E)(y_1, \ldots, y_n) \approx E[y_1/x_1, \ldots, y_n/x_n])$

$\forall \bar{x}(\Delta(\mu X . M)^i(\bar{x}) \rightarrow (\mu X . M)(\bar{x}) \approx (\mu X . M)^i(\bar{x}))$ $(i<\omega)$

siendo χ una fórmula libre de cuantificadores (y, en ciertos casos, libre del operador de definición Δ, según lo exija la sintaxis), $f \in \overline{\Sigma}$, un símbolo de función, $X \in VF^{\tau}$, una variable funcional de tipo τ, y $\bar{x}$ (idem $\bar{y}$),

una n-tupla de variables distintas dos a dos $x_1..x_n$, con lo que $\forall\bar{x}$ abrevia $\forall x_1..\forall x_n$ y $\bar{x}=\bar{y}$, $x_1=y_1\wedge..\wedge x_n=y_n$. Así mismo entiéndase que con $\chi(\bar{x})$ indicamos que $lib(\chi)\subseteq\{x_1,..,x_n\}$, donde $\bar{x}$ es una n-tupla de variables distintas dos a dos. Aunque no se dice, se supone que existe coherencia de tipos entre todas los símbolos que aparecen en las fórmulas.

Un axioma de la igualdad se dice adecuado a un conjunto de fórmulas Ψ si solo usa constantes, símbolos de función y símbolos de predicado que aparezcan en Ψ, y si además todas sus variables libres aparecen libres en alguna fórmula de Ψ.

Dado un árbol ω-ramificado Σ escribimos $\Sigma \xrightarrow[IG]{} \Sigma'$ para representar que Σ' es el árbol que se obtiene extendiendo todas las ramas abiertas de Σ mediante el axioma de la igualdad θ a través del siguiente proceso:

Dada una rama abierta de Σ con Ψ como conjunto de fórmulas etiquetando su hoja, si θ es adecuado a Ψ entonces ampliar la rama con un nuevo nodo etiquetado con $\Psi \cup \{\theta\}$.

En las mismas condiciones, dada una fórmula φ, escribimos $\Sigma \xrightarrow[\varphi]{} \Sigma'$ para representar que Σ' es el árbol que se obtiene extendiendo todas las ramas abiertas de Σ mediante el siguiente proceso:

Dada una rama abierta de Σ con Ψ como conjunto de fórmulas etiquetando su hoja entonces ampliar la rama con un nuevo nodo etiquetado con $\Psi \cup \{\varphi\}$.

Dados los árboles ω-ramificados Σ, Σ', decimos que Σ' es una extensión de Σ, lo que escribimos mediante $\Sigma \rightarrow \Sigma'$, si se da alguno de los siguientes casos, o bien $\Sigma \xrightarrow[B]{} \Sigma'$ o $\Sigma \xrightarrow[IG]{} \Sigma'$ o $\Sigma \xrightarrow[\varphi]{} \Sigma'$.

Un tableau de ramas finitas para un conjunto de $\bar{\Sigma}$-fórmulas Φ es un árbol definido inductivamente por las siguientes reglas:

(INI) Si $\Phi_0 \subseteq \Phi$ entonces el árbol:

$$\circ\ \Phi_0$$

es un tableau de ramas finitas para Φ.

(ET) Si Σ es un tableau de ramas finitas para Φ y $\Sigma \rightarrow \Sigma'$ entonces el árbol Σ' es un tableau de ramas finitas para Φ. ■

Dada la definición de tableaux de ramas finitas podemos definir sucesiones de ellos tal como se hizo en el capítulo anterior. Supuesta entonces una noción de límite de una sucesión de tableaux de ramas finitas, similar a la dada en §2.4, se puede dar la definición general de tableau.

3.-Definición. *Sucesiones de tableaux. Tableaux.*

Dado un conjunto de $\widetilde{\Sigma}$-fórmulas Φ, decimos que $\langle \mathcal{I}_k \ / \ k \in \mathbb{N} \rangle$, es una sucesión de tableaux de ramas finitas para Φ si:

(i) $\mathcal{I}_0$ es un tableau de ramas finitas como los obtenidos por (INI) de la definición anterior.

(ii) $\mathcal{I}_k \longrightarrow \mathcal{I}_{k+1}$, para todo $k \in \mathbb{N}$.

Un tableau para un conjunto de fórmulas Φ es o bien un tableau de ramas finitas para Φ o bien el límite de una sucesión de tableaux de ramas finitas para Φ. ■

Razones que se harán precisas más adelante obligan a que los tableaux de ramas finitas satisfagan la propiedad de que los hijos de la raíz de uno cualquiera de ellos sean a su vez tableaux de ramas finitas. Obsérvese que esto era evidente para los tableaux de LFP debido a la finitud de todos los procesos de construcción de tableaux haciendo que éstos se puedan obtener tanto de la raíz a las hojas como de éstas a la raíz. En LFRP los árboles son ω-ramificados y, por tanto, el proceso de "abajo a arriba" no es inmediato que dé lugar a todos los tableaux de ramas finitas. Es por esto que demostramos el siguiente resultado.

4.-Lema.

El árbol $\mathcal{I}$ es un tableau de ramas finitas para un conjunto de fórmulas Φ si y solo si se da alguna de las siguientes condiciones:

(a) $\mathcal{I}$ es un árbol cuya raíz está etiquetada con un conjunto $\Psi(\subseteq\Phi)$

(b) existe un subconjunto Ψ de Φ tal que Ψ es coherente y $\mathcal{I}$ se ha obtenido de alguna de las siguientes maneras:

(i) existe una fórmula conjuntiva α (resp. ω-conjuntiva $\alpha^{\bullet}$) en Ψ tal que α_1 (resp. $\alpha_1^{\bullet}$) es un constituyente de α (resp. $\alpha_1^{\bullet}$) que no pertenece a Ψ y existe un tableau de ramas finitas $\mathcal{I}'$ para un conjunto de fórmulas que incluye a $\Psi \cup \{\alpha_1\}$ (resp. $\Psi \cup \{\alpha_1^{\bullet}\}$) de forma que $\mathcal{I} =$ **Alargar**$(\mathcal{I}', \Psi)$

(ii) existe una fórmula disyuntiva β (resp. ω-disyuntiva $\beta^{\bullet}$) en Ψ y m tableaux de ramas finitas $\mathcal{I}_i$ para un conjunto de fórmulas que incluye a $\Psi \cup \{\beta_i\}$, $1 \leq i \leq m$, (resp. ω tableaux para $\Psi \cup \{\beta_1^{\bullet}\}$), de forma que $\mathcal{I} =$ **Ramificar**$(\{\mathcal{I}_i \ / \ 1 \leq i \leq m$ (resp. $i < \omega)\}, \Psi)$

(iii) existe una fórmula universal γ en Ψ, una expresión E adecuada a Ψ y un tableau de ramas finitas $\mathcal{I}'$ para un conjunto de fórmulas que incluye a $\Psi \cup \{\gamma(E)\}$ de forma que $\mathcal{I} =$ **Alargar**$(\mathcal{I}', \Psi)$

(iv) existe una fórmula existencial δ en Ψ, una constante auxiliar c que no aparece en Ψ y un tableau de ramas finitas $\mathcal{T}'$ para un conjunto de fórmulas que incluye a $\Psi \cup \{\delta(c)\}$ de forma que $\mathcal{T}$ = **Alargar**($\mathcal{T}'$, Ψ)

(v) existe una fórmula $\varphi \in \Phi$ y un tableau de ramas finitas $\mathcal{T}'$ para un conjunto de fórmulas que incluye a $\Psi \cup \{\varphi\}$ de forma que $\mathcal{T}$ = **Alargar**($\mathcal{T}'$, Ψ)

(vi) existe un axioma de la igualdad θ adecuado a Ψ y un tableau de ramas finitas $\mathcal{T}'$ para un conjunto de fórmulas que incluye al dado por $\Psi \cup \{\theta\}$ de forma que $\mathcal{T}$ = **Alargar**($\mathcal{T}'$, Ψ)

donde **Alargar**($\mathcal{T}$, Ψ) es el árbol formado al poner el árbol $\mathcal{T}$, como hoja de otro árbol formado por un solo nodo etiquetado por Ψ, y **Ramificar**($\{\mathcal{T}_i$ / $1 \le i \le m$ (resp. $i < \omega$)$\}$, Ψ), es el obtenido al poner como hojas, de forma similar, los árboles $\mathcal{T}_i$, $1 \le i \le m$ (resp. $i < \omega$).

Demostración. Si $\mathcal{T}$ es un tableau de ramas finitas para el conjunto de fórmulas Φ entonces o bien se ha obtenido por la regla (INI) de 2, y se da (a), o por la regla (ET), también de 2. En este caso, supongamos que existe un solo hijo para la raíz Ψ de $\mathcal{T}$ y sea el árbol $\mathcal{T}'$ lo que nace de tal hijo; entonces $\mathcal{T}$ = **Alargar**($\mathcal{T}'$, Ψ).

Es fácil demostrar que $\mathcal{T}'$ es un tableau de ramas finitas para un conjunto de fórmulas que incluye a la etiqueta Ψ' de su raíz. El conjunto Ψ' depende de la forma en que se pasó de Ψ a Ψ'; por ejemplo, si fue por el constituyente de una fórmula conjuntiva α entonces $\alpha \in \Psi$, $\alpha_1 \notin \Psi$ y $\Psi' = \Psi \cup \{\alpha_1\}$. Obsérvese que esto realmente ocurre así debido a que las fórmulas que etiquetan un nodo pasan, cuando se extiende, a su hijo; en efecto, si en lugar de etiquetar nodos con conjuntos lo hiciéramos con fórmulas, al suprimir la raíz, en nuestro caso una fórmula conjuntiva, un paso posterior en $\mathcal{T}'$ que añadiera un nuevo constituyente de α no quedaría justificado pues α ya no está.

De forma análoga se razona si existen ω o m hijos. En consecuencia, se da alguna de las condiciones (i)-(vi) de (b), como queríamos demostrar.

El recíproco del lema en el caso (a) es evidente. Para (b) si, por ejemplo, se ha dado el caso (i), esto es, $\mathcal{T}$ = **Alargar**($\mathcal{T}'$, Ψ) por una fórmula conjuntiva, es fácil demostrar que $\mathcal{T}$ es un tableau de ramas finitas para Φ, para lo cual se tiene en cuenta que la primera extensión se debe al análisis, mediante (A) de 2, del nodo etiquetado con Ψ, y el resto de las extensiones se producen como lo hicieron para $\mathcal{T}'$. ■

Finalizamos esta sección con un ejemplo que muestre el uso del método de los tableaux que acabamos de definir.

Consideremos la expresión funcional $M = \lambda y.\, y \simeq 0 \longrightarrow 1, (y \simeq 1 \longrightarrow X(1),\ X(y-2))$ sobre la signatura Σ de género $S=\{nat\}$ y los conjuntos de símbolos $C_{\Sigma}^{nat}=\{0\}$, $SF_{\Sigma}^{nat\longrightarrow nat}=\{suc\}$ y $SF_{\Sigma}^{nat,nat\longrightarrow nat}=\{-\}$. Entendemos que n es el Σ-término $suc(suc(\overset{n}{\ldots}(suc(0))\ldots))$.

Veamos que el conjunto de fórmulas $\Phi = \{\neg(1\simeq 0),\ \Delta(\mu X.M)(1)\}$ tiene un tableau de ramas finitas $\mathcal{T}$ tal que todas sus ramas están cerradas, lo que como veremos supone que lo esté también el tableau. Como en el capítulo anterior esto implicará que el conjunto no sea fuertemente satisfactible y, según §1.B.2.2.(a), tendremos que $\neg(1\simeq 0) \vdash \neg\Delta(\mu X.M)(1)$.

El primer tableau lo constituye un nodo etiquetado con el conjunto de fórmulas Φ. Esto es:

$$\odot\ \Phi$$

Analizando este nodo de acuerdo con $(B^{\bullet})$, lo bifurcamos creando ω nuevos nodos de la siguiente manera:

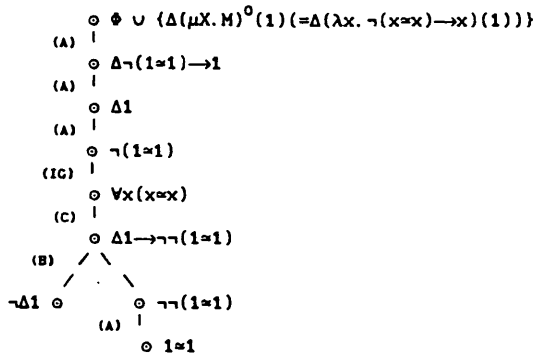
$$\odot\ \Phi$$

$$\odot\ \Phi \cup \{\Delta(\mu X.M)^{0}(1)\} \qquad \odot\ \Phi \cup \{\Delta(\mu X.M)^{1}(1)\} \qquad \ldots$$

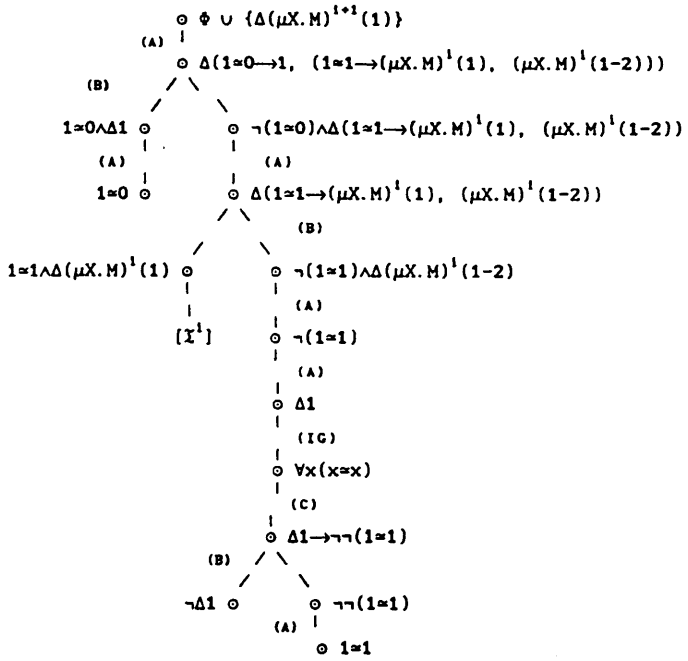
La continuación del tableau la haremos construyendo un tableau de ramas finitas para cada una de las ω hojas anteriores. Representemos por $\mathcal{T}^{i}$ el tableau que tiene por raíz a $\Phi \cup \{\Delta(\mu X.M)^{i}(1)\}$, $i<\omega$. Las ramas de cada uno de estos tableaux irán etiquetadas con una marca de entre (A), (B), (C), (D), $(A^{\bullet})$, $(B^{\bullet})$ e (IG). Dichas marcas supondrían el análisis del nodo origen de la rama etiquetada, que en el momento del análisis sería hoja, e implicarían que se ha hecho una extensión del tableau $\mathcal{T}$ de la forma $\mathcal{T} \xrightarrow[B]{} \mathcal{T}'$, para las seis primeras, siendo B el conjunto de ramas formado únicamente por la rama extendida, o $\mathcal{T} \xrightarrow[IG]{} \mathcal{T}'$, para la última. En beneficio de la escritura supondremos, por una parte, que cada nodo va etiquetado solamente con la fórmula nueva introducida, y por otra, que la extensión de una rama mediante un axioma de la igualdad afecta solo a la rama para quien se introduce dicho axioma. Una vez construidos los tableaux $\mathcal{T}^{i}$, $i<\omega$, el lema anterior demostrará que el tableau obtenido añadiendo, al que acabamos de mencionar, los $\mathcal{T}^{i}$ como árboles en sus hojas, es un tableau para Φ.

Los tableaux Υ^i se construyen por inducción sobre i. Se tiene que:

(i) Υ^0:



(i) Paso de Υ^i a Υ^{i+1}:



donde $[\Upsilon^i]$ es el árbol Υ^i, pero suponiendo que a los conjuntos de fórmulas que etiquetan sus nodos se añaden las fórmulas que han ido apareciendo en la

construcción de la rama de Σ^{i+1} que termina en $[\Sigma^i]$.

Una vez que se añaden los árboles Σ^i en los nodos $\Phi \cup \{\Delta(\mu X.M)^i(1)\}$ que mencionábamos más arriba, es fácil demostrar que todas las ramas de Σ^i están cerradas, para todo $i<\omega$. En efecto, las de Σ^0 es evidente que lo están, mientras que Σ^{i+1} también lo está si lo estaba Σ^i, pues la rama que termina en $\neg(1\approx 0)$ "se cierra" con la fórmula $1\approx 0$ de la raíz. Esto demuestra obviamente que todas las ramas de Σ están cerradas.

8.- CORRECCION DEL METODO DE LOS TABLEAUX.

La corrección y completitud del método de los tableaux que acabamos de introducir tiene el mismo sentido que para la lógica LFP. Se trata de demostrar la equivalencia entre la inexistencia de un modelo fuerte para un conjunto de fórmulas y la existencia para él de un tableau cerrado. En este caso, este último concepto no se entiende como en el capítulo anterior. Recuérdese que entonces todos los tableaux cerrados eran tableaux finitos. Ahora un tableau cerrado puede no ser de profundidad finita; por ejemplo, el tableau $\mathcal{T}$, obtenido en el ejemplo anterior, es cerrado y, sin embargo, $d(\mathcal{T}) = \bigcup d(\mathcal{T}^i) = \omega$, pues es fácil comprobar que:

(i) $0 < d(\mathcal{T}^i) < \omega$, para todo $i<\omega$

(ii) $d(\mathcal{T}^i) < d(\mathcal{T}^{i+1})$, para todo $i<\omega$.

Sí exigiremos, en cambio, que las ramas de un tableau cerrado se hayan cerrado en un número finito de pasos.

1.-Definición. *Tableaux abiertos y cerrados.*

Un tableau $\mathcal{T}$ se dice cerrado si es un tableau de ramas finitas en el que todas ellas están cerradas. Se dice abierto si no es cerrado. ■

La construcción de un tableau de ramas finitas para un conjunto de fórmulas verifica que si el conjunto es satisfactible, al menos una rama del tableau está etiquetada por un conjunto también satisfactible.

2.-Lema.

Sea $\mathcal{T}$ un tableau de ramas finitas para un conjunto de Σ-fórmulas Φ y sea $\Phi_0 \subseteq \Phi$, el subconjunto de Φ usado en la construcción de $\mathcal{T}$. Para toda Σ-interpretación $\mathcal{I}$:

$\mathcal{I}$ es modelo fuerte de Φ_0 ⇔ existen una rama (finita) de $\mathcal{T}$ con Ψ_0 como conjunto de fórmulas etiquetando su hoja y una $\bar{\Sigma}$-interpretación $\bar{\mathcal{I}}$ expandiendo $\mathcal{I}$ tales que $\bar{\mathcal{I}}$ es modelo fuerte de Ψ_0.

Demostración. (⇒) Por inducción sobre la construcción de $\mathcal{T}$. Si se ha obtenido por aplicación de la regla (INI) de 7.2 entonces es evidente pues $\mathcal{T}$ tiene una única rama formada por un solo nodo etiquetado con un conjunto de fórmulas $\Phi_0 \subseteq \Phi$ y, por hipótesis, $\mathcal{I}$ es modelo fuerte de Φ.

Si Σ se ha obtenido por aplicación de la regla (ET) de 7.2 entonces existe un tableau de ramas finitas Σ' tal que $\Sigma' \longrightarrow \Sigma$. Por hipótesis de inducción existe una rama (finita) R' en Σ' y una expansión $\overline{\mathfrak{I}}'$ de $\mathfrak{I}$ tales que $\overline{\mathfrak{I}}'$ es modelo fuerte del conjunto de fórmulas Ψ_0' que etiqueta la hoja de R'.

Si R' no se ha prolongado en la extensión de Σ', es inmediato. Si no, tomemos como R la prolongación de R' y distingamos casos según haya sido la extensión del tableau Σ'.

Supongamos pues, en primer lugar, que la extensión se ha producido por análisis de un conjunto de ramas B. Como R' se ha prolongado entonces $R' \in B$. Demostremos que la extensión R de R' es una rama abierta de Σ para la que existe una expansión $\overline{\mathfrak{I}}'$ de $\mathfrak{I}$ que es modelo fuerte del conjunto de fórmulas Ψ_0 que etiqueta su hoja.

Para demostrarlo distinguimos casos según la regla (A)-(B^*) de 7.2 empleada para la extensión. Para (A), (B), (C), (D) se sigue una demostración similar a la de §2.5.3. Respecto a (A^*) el resultado se sigue de forma inmediata de la condición ω-ALFA vista en 6.2, tomando la misma $\overline{\mathfrak{I}}'$ que existe, por hipótesis de inducción. De forma similar para (B^*), usando la condición ω-BETA. Por último decir que, evidentemente, si $\overline{\mathfrak{I}}$ es modelo fuerte de las fórmulas que etiquetan la hoja de R entonces R no puede estar cerrada, pues los conjuntos incoherentes de fórmulas no son fuertemenete satisfactibles.

Si la extensión de Σ se ha producido prolongando todas las ramas abiertas con un nuevo nodo etiquetado con un axioma de la igualdad, razonando similarmente al caso anterior basta demostrar que la interpretación $\overline{\mathfrak{I}}'$, que existe por hipótesis de inducción, es modelo fuerte del axioma. Más en general, como en el capítulo anterior, demostremos que los axiomas de la igualdad son verdaderos en toda $\overline{\Sigma}$-interpretación $\overline{\mathfrak{I}}$. Para ello solo nos ocuparemos de los nuevos tipos de axiomas, remitiendo a §2.5.3 para el resto.

El caso para $\forall\bar{x}\forall\bar{y}(\bar{x}\approx\bar{y} \wedge \Delta X(\bar{x}) \longrightarrow \Delta X(\bar{y}))$ es análogo a $\forall\bar{x}\forall\bar{y}(\bar{x}\approx\bar{y} \wedge \Delta f(\bar{x}) \longrightarrow \Delta f(\bar{y}))$. Para el axioma:

$$\forall\bar{y}(\Delta E[y_1/x_1,\ldots,y_n/x_n] \longrightarrow (\lambda x_1..x_n.E)(\bar{y}) \approx E[y_1/x_1,\ldots,y_n/x_n])$$

entendiendo que la tupla $\bar{y}$ se refiere a $(y_1,\ldots,y_n)$, basta demostrar que para toda n-tupla $\bar{a}$ cualquiera sobre el dominio de $\overline{\mathfrak{I}}$, se tiene que:

$$\overline{\mathfrak{I}}[\bar{a}/\bar{y}] \models_s \Delta E[y_1/x_1,\ldots,y_n/x_n] \longrightarrow (\lambda x_1..x_n.E)(\bar{y}) \approx E[y_1..y_n/x_1..x_n]$$

Si $\overline{\mathfrak{I}}[\bar{a}/\bar{y}](\Delta E[y_1/x_1,\ldots,y_n/x_n]) = \underline{f}$, no hay nada que demostrar. Supongamos pues que $\overline{\mathfrak{I}}[\bar{a}/\bar{y}](\Delta E[y_1/x_1,\ldots,y_n/x_n]) = \underline{t}$ entonces:

$\mathfrak{I}[\bar{a}/\bar{y}](E[y_1/x_1,\ldots,y_n/x_n]) \neq \perp_A$.

Pero $\mathfrak{I}[\bar{a}/\bar{y}]((\lambda x_1..x_n.E)(\bar{y})) = \mathfrak{I}[\bar{a}/\bar{y}](\lambda x_1..x_n.E)(\bar{a}) =$

$= \mathfrak{I}[\bar{a}/\bar{y}][\bar{a}/\bar{x}](E) =$

$= \mathfrak{I}[\bar{a}/\bar{y}](E[\bar{y}/\bar{x}])$ (por el lema de sustitución 4.5)

luego:

$$\mathfrak{I}[\bar{a}/\bar{y}]((\lambda x_1..x_n.E)(\bar{y}) \simeq E[\bar{y}/\bar{x}])) = \underline{t}$$

como queríamos demostrar.

Similarmente, para $\forall\bar{x}(\Delta(\mu X.M)^i(\bar{x}) \rightarrow (\mu X.M)(\bar{x}) \simeq (\mu X.M)^i(\bar{x}))$, $(i<\omega)$, si $\mathfrak{I}[\bar{a}/\bar{x}](\Delta(\mu X.M)^i(\bar{x})) = \underline{t}$ entonces $\mathfrak{I}[\bar{a}/\bar{x}]((\mu X.M)^i(\bar{x})) \neq \perp_A$ luego $\mathfrak{I}[\bar{a}/\bar{x}]((\mu X.M)(\bar{x})) = \bigcup\{\mathfrak{I}[\bar{a}/\bar{x}]((\mu X.M)^j(\bar{x}))/j<\omega\}$ (por 5.4) $= \mathfrak{I}[\bar{a}/\bar{x}]((\mu X.M)^i(\bar{x}))$ (por tratarse de cpo's planos) y entonces $\mathfrak{I}[\bar{a}/\bar{x}]((\mu X.M)(\bar{x}) \simeq (\mu X.M)^i(\bar{x})) = \underline{t}$.

Por último, si la extensión de $\mathcal{I}$ se ha producido prolongando todas las ramas abiertas con un nuevo nodo etiquetado con una fórmula del conjunto Φ, la demostración es evidente tomando la misma $\mathfrak{I}'$, que existe por hipótesis de inducción, pues ésta interpretación expande $\mathfrak{I}$ e $\mathfrak{I}$ es modelo fuerte de Φ.

(*) Es evidente considerando la restricción de $\mathfrak{I}$ a la signatura Σ y teniendo en cuenta que, por construcción, $\Psi_0 \supseteq \Phi_0$. ■

3.-Teorema. *Corrección del método de los tableaux.*

Si un conjunto de fórmulas tiene un tableau cerrado entonces no es fuertemente satisfactible.

Demostración. Sea $\mathcal{I}$ un tableau cerrado para Φ. Por definición de tableau cerrado, $\mathcal{I}$ es un tableau de ramas finitas y cerradas. Si Φ fuese fuertemente satisfactible entonces también lo sería Φ_0, el subconjunto de Φ usado en la construcción de $\mathcal{I}$. Sea $\mathfrak{I}$ un modelo de Φ_0 entonces, por el lema anterior, existe una expansión $\bar{\mathfrak{I}}$ de $\mathfrak{I}$ y una rama (finita) en $\mathcal{I}$ con Ψ_0 como conjunto de fórmulas etiquetando su hoja tales que $\bar{\mathfrak{I}}$ es modelo fuerte de Ψ_0. Ahora bien, como $\mathcal{I}$ es cerrado todas sus ramas son cerradas luego Ψ_0 es un conjunto incoherente de fórmulas y, por tanto, no satisfactible fuertemente, contra lo supuesto. En conclusión, Φ no lo es tampoco. ■

9.- COMPLETITUD DEL METODO DE LOS TABLEAUX.

El concepto de conjunto de Hintikka que ya introdujimos en el capítulo anterior para demostrar la completitud del método de los tableaux, será ahora extendido para poder tratar las fórmulas ω-conjuntivas y ω-disyuntivas.

1.-Definición. *Conjunto de Hintikka.*

Un conjunto H de $\overline{\Sigma}$-fórmulas se dice que es un conjunto de Hintikka si y solo si satisface las siguientes condiciones:

(A) Para toda fórmula $\alpha\in$ALFA: $\alpha\in H \Rightarrow \alpha_i\in H$, para todo $i=1,..,m$.

(B) Para toda fórmula $\beta\in$BETA: $\beta\in H \Rightarrow \beta_i\in H$, para algún $i=1,..,m$.

(C) Para toda fórmula $\gamma\in$GAMMA: $\gamma\in H \Rightarrow \gamma(E)\in H$, para toda $\overline{\Sigma}$-expresión E de género σ, adecuada a H.

(D) Para toda fórmula $\delta\in$DELTA: $\delta\in H \Rightarrow \delta(c)\in H$, para alguna $c\in C^{\sigma}$ de género σ adecuado.

(A^*) Para toda fórmula $\alpha^*\in\omega$-ALFA: $\alpha^*\in H \Rightarrow \alpha^*_i\in H$, para todo $i<\omega$.

(B^*) Para toda fórmula $\beta^*\in\omega$-BETA: $\beta^*\in H \Rightarrow \beta^*_i\in H$, para algún $i<\omega$.

(E) Todo axioma de la igualdad adecuado a H pertenece a H.

(F) H es un conjunto coherente de fórmulas. ■

En el apartado (C) (resp. (D)) de la definición anterior se supone que la expresión E (resp. c) tiene género σ, entendiendo que la fórmula universal γ es de la forma $\neg\exists x^{\sigma}\varphi$ (resp. $\exists x^{\sigma}\varphi$) y que, por tanto, la sustitución está bien definida.

Como los conjuntos de Hintikka para la lógica de primer orden, los que acabamos de definir satisfacen la propiedad de que aquellas expresiones adecuadas acerca de las cuales no se afirma nada sobre su indefinición, puede concluirse que están definidas.

2.-Lema.

Dados un conjunto de Hintikka H y una expresión $E\in E^{\sigma}_{\overline{\Sigma}}$ adecuada a él, se tiene que: $\neg\Delta E\in H \Rightarrow \Delta E\in H$.

Demostración. Como en §2.6.2. ■

En la demostración de la satisfactibilidad fuerte de todo conjunto de Hintikka nos centraremos en aquellos casos particulares de la lógica LFRP, remitiendo el resto a lo visto para LFP.

3.-Teorema. *Satisfactibilidad fuerte de los conjuntos de Hintikka.*

Todo conjunto de Hintikka es fuertemente satisfactible.

Demostración. Sea $H \subseteq F_{\overline{\Sigma}}$ un conjunto de Hintikka y, para cada género $\sigma \in S$, consideremos el conjunto de $\overline{\Sigma}$-expresiones de género σ:

$$T_{(H,\sigma)} = \{E \in E_{\overline{\Sigma}}^{\sigma} \ / \ \Delta E \in H\}$$

El conjunto $T_{(H,\sigma)}$ es no vacío por las mismas razones ya vistas para LFP. Sobre $T_{(H,\sigma)} \times T_{(H,\sigma)}$ consideramos la relación binaria $\approx_{\sigma}$ definida por:

$$E \approx_{\sigma} E' \Leftrightarrow E \simeq_{\sigma} E' \in H$$

Como vimos en el capítulo anterior, es posible demostrar que $\approx_{\sigma}$ es una relación de equivalencia, usando los axiomas de la igualdad que expresan la reflexividad, simetría y transitividad de $\simeq_{\sigma}$. También se puede demostrar que $\approx_{\sigma}$ es congruente con respecto a símbolos de función, relación y variables funcionales, en el sentido que se vio para los dos primeros y que precisamos a continuación.

Dados E_i, $E'_i \in T_{(H,\sigma i)}$ tales que $E_i \approx_{\sigma i} E'_i$, $i=1,\ldots,n$, se tiene:

(i) Para todo símbolo de función $f \in SF_{\overline{\Sigma}}^{\tau}$, $\tau = \sigma_1 \ldots \sigma_n \rightarrow \sigma$:

$$\Delta f(E_1,\ldots,E_n) \in H \Leftrightarrow \Delta f(E'_1,\ldots,E'_n) \in H$$

(ii) Para toda fórmula $\chi(x_1,\ldots,x_n)$ libre de cuantificadores y símbolos del operador de definición Δ:

$$\chi(E_1,\ldots,E_n) \in H \Leftrightarrow \chi(E'_1,\ldots,E'_n) \in H$$

(iii) Para toda variable funcional $X \in VF^{\tau}$, $\tau = \sigma_1 \ldots \sigma_n \rightarrow \sigma$:

$$\Delta X(E_1,\ldots,E_n) \in H \Leftrightarrow \Delta X(E'_1,\ldots,E'_n) \in H$$

La demostración de (iii) es análoga a la de (i), usando el axioma: $\forall \overline{x} \forall \overline{y} (\overline{x} \approx \overline{y} \wedge \Delta X(\overline{x}) \rightarrow \Delta X(\overline{y}))$.

De lo que acabamos de demostrar podemos deducir que la siguiente $\overline{\Sigma}$-interpretación está bien definida. Consideremos $\overline{\mathfrak{I}} = (\overline{\mathfrak{A}}, \varepsilon)$ dada por:

(a) Conjuntos soporte de género σ:

$$A^{\sigma} = T_{(H,\sigma)}/\approx_{\sigma}$$

Representaremos mediante $[E]$, la clase de equivalencia de E.

(b) Interpretación de las constantes $c \in C_{\Sigma}^{\sigma}$ de género σ:

$$c^{\overline{\mathfrak{A}}} = \begin{cases} [c] & \text{si } \Delta c \in H \\ \perp_{\sigma} & \text{en otro caso} \end{cases}$$

(c) Interpretación de las constantes auxiliares $c \in C^{\sigma}$ de género σ:

$$c^{\overline{\mathfrak{A}}} = \begin{cases} [c] & \text{si } c \text{ aparece en } H \\ \text{elemento arbitrario de } A^{\sigma} & \text{en otro caso} \end{cases}$$

(d) Interpretación de los símbolos de función $f \in SF^{\tau}_{\Sigma}$, $\tau = \sigma_1 .. \sigma_n \rightarrow \sigma$:

$$f^{\overline{\mathfrak{A}}}([E_1], \ldots, [E_n]) = \begin{cases} [f(E_1, \ldots, E_n)] & \text{si } \Delta f(E_1, \ldots, E_n) \in H \\ \perp_{\sigma} & \text{en otro caso} \end{cases}$$

para todos $[E_i] \in A^{\sigma_i}$, $i=1,..,n$, y:

$$f^{\overline{\mathfrak{A}}}(\perp_{\otimes}) = \perp_{\sigma}$$

donde $\perp_{\otimes}$ es el elemento "bottom" del producto cartesiano estricto.

(e) Interpretación de los símbolos de relación $R \in SR^{\rho}_{\Sigma}$, $\rho = \sigma_1 .. \sigma_n$:

$$R^{\overline{\mathfrak{A}}}([E_1], \ldots, [E_n]) = \begin{cases} \underline{t} & \text{si } R(E_1, \ldots, E_n) \in H \\ \underline{f} & \text{si } \neg R(E_1, \ldots, E_n) \in H \\ \perp & \text{si } \neg \Delta R(E_1, \ldots, E_n) \in H \\ \text{arbitrario} & \text{en otro caso} \end{cases}$$

para todos $[E_i] \in A^{\sigma_i}$, $i=1,..,n$, y:

$$R^{\overline{\mathfrak{A}}}(\perp_{\otimes}) = \perp.$$

(f) Valoración de variables de individuo $x \in V^{\sigma}$, $\sigma \in S$:

$$\varepsilon(x) = \begin{cases} [x] & \text{si } \Delta x \in H \\ \text{elemento arbitrario de } A^{\sigma} & \text{en otro caso} \end{cases}$$

(g) Valoración de variables funcionales $X \in VF^{\tau}$, $\tau = \sigma_1 .. \sigma_n \rightarrow \sigma$:

$$\varepsilon(X)([E_1], \ldots, [E_n]) = \begin{cases} [X(E_1, \ldots, E_n)] & \text{si } \Delta X(E_1, \ldots, E_n) \in H \\ \perp_{A} & \text{en otro caso} \end{cases}$$

para todos $[E_i] \in A^{\sigma_i}$, $i=1,..,n$, y:

$$\varepsilon(X)(\perp_{\otimes}) = \perp_{\sigma}.$$

Obsérvese que una variable funcional libre X se trata de la misma forma que los símbolos de función, a efectos de la construcción de modelo.

Una vez definida la $\overline{\Sigma}$-interpretación $\overline{\mathfrak{I}}$ veamos que es modelo fuerte de todas las fórmulas de H. Para ello demostramos por inducción simultánea sobre E y φ (inducción que se basará sobre una medida de complejidad que demuestra que cada fórmula es más compleja que sus constituyentes), los siguientes tres hechos:

Hecho 3.1: Para toda expresión $E \in E^{\sigma}_{\overline{\Sigma}}$, $\sigma \in S$: $\Delta E \in H \Rightarrow \overline{\mathfrak{I}}(E) = [E]$

Hecho 3.2: Para toda expresión $E \in E^{\sigma}_{\overline{\Sigma}}$, $\sigma \in S$: $\neg \Delta E \in H \Rightarrow \overline{\mathfrak{I}}(E) = \perp_{\sigma}$

Hecho 3.3: Para toda fórmula $\varphi \in F_{\overline{\Sigma}}$: $\varphi \in H \Rightarrow \overline{\mathfrak{I}}(\varphi) = \underline{t}$.

Demostración del Hecho 3.1.

Si E es o bien un término, o bien de la forma $\square(E_1,\ldots,E_n)$, $\square\in\{f, X\}$ o bien una expresión condicional de la forma $\chi\longrightarrow E$ o $\chi\longrightarrow E, E'$, basta seguir una demostración similar a la dada en §2.6.3.1. Para el resto de los casos tenemos lo siguiente.

(i) $\Delta(\lambda x_1..x_n.E)(E_1,\ldots,E_n)\in H \Rightarrow \Delta E[E_1/x_1,\ldots,E_n/x_n],\ \Delta E_1,\ldots,\ \Delta E_n\in H$ (A)
$\Rightarrow\ \overline{\mathfrak{I}}(E[E_1/x_1,\ldots,E_n/x_n]) = [E[E_1/x_1,\ldots,E_n/x_n]]$ (por HI; léase "hipótesis de inducción")

Ahora bien, por la condición (E), el axioma:

$\forall y_1..\forall y_n(\Delta E[y_1/x_1,\ldots,y_n/x_n]\longrightarrow(\lambda x_1..x_n.E)(y_1,\ldots,y_n)\simeq E[y_1/x_1,\ldots,y_n/x_n])$

pertenece a H. Particularizando para $E_1,\ldots,E_n$, respectivamente, y teniendo en cuenta que $\Delta E_i\in H$, i=1,..,n, es fácil ver que:

$$\Delta E[E_1/x_1,\ldots,E_n/x_n]\longrightarrow(\lambda x_1..x_n.E)(E_1,\ldots,E_n)\simeq E[E_1/x_1,\ldots,E_n/x_n]\in H$$

y como acabamos de ver que $\Delta E[E_1/x_1,\ldots,E_n/x_n]\in H$, es inmediato entonces que:

$$(\lambda x_1..x_n.E)(E_1,\ldots,E_n)\simeq E[E_1/x_1,\ldots,E_n/x_n]\in H$$

con lo que:

$$[(\lambda x_1..x_n.E)(E_1,\ldots,E_n)] = [E[E_1/x_1,\ldots,E_n/x_n]]$$

pues recuérdese que $\Delta(\lambda x_1..x_n.E)(E_1,\ldots,E_n)$, $\Delta E[E_1/x_1,\ldots,E_n/x_n]\in H$ y, por tanto, ambas clases tienen sentido.

En conclusión:

$$\begin{aligned}\overline{\mathfrak{I}}((\lambda x_1..x_n.E)(E_1,..,E_n)) &= \overline{\mathfrak{I}}(\lambda x_1..x_n.E)(\overline{\mathfrak{I}}(E_1),..,\overline{\mathfrak{I}}(E_n)) = \\ &= \overline{\mathfrak{I}}(\lambda x_1..x_n.E)([E_1],\ldots,[E_n])\ \text{(pues } \Delta E_i\in H,\ i=1,..,n,\\ &\qquad \text{luego } \overline{\mathfrak{I}}(E_i)=[E_i],\ i=1,..,n,\ \text{por HI)}\\ &= \overline{\mathfrak{I}}[[E_1]/x_1,\ldots,[E_n]/x_n](E)\ \text{(por sem. de abstracción)}\\ &= \overline{\mathfrak{I}}(E[E_1/x_1,\ldots,E_n/x_n])\ \text{(por lema de sustitución 4.5)}\\ &= [E[E_1/x_1,\ldots,E_n/x_n]]\\ &= [(\lambda x_1..x_n.E)(E_1,\ldots,E_n)]\end{aligned}$$

como queríamos demostrar.

(ii) $\Delta(\mu X.M)(E_1,\ldots,E_n)\in H \Rightarrow \Delta(\mu X.M)^i(E_1,\ldots,E_n)\in H$, para algún $i<\omega$ (B^*)

Dada una expresión funcional $M\in EF_{\overline{\Sigma}}$, se puede demostrar que si se tiene $\Delta M(E_1,\ldots,E_n)\in H$ entonces $\Delta E_i\in H$, para todo i=1,..,n. En efecto, demostrémoslo por inducción sobre M. Para M siendo f o X es evidente por (A), usando (α7) y (α8) de la clasificación orientada hacia los tableaux. Para una abstracción usamos de la misma forma (α9). Para una recursión tenemos que si $\Delta(\mu X.M)(E_1,\ldots,E_n)\in H$ entonces $\Delta(\mu X.M)^1(E_1,\ldots,E_n)\in H$, para algún $i<\omega$, por (β^*), luego, por hipótesis de inducción para $(\mu X.M)^1$, resulta que $\Delta E_i\in H$, i=1,..,n, como queríamos demostrar.

De lo que acabamos de decir deducimos que $\Delta E_i \in H$, para todo $i=1,\ldots,n$. Usando que el axioma:

$$\forall \bar{x}(\Delta(\mu X.M)^i(\bar{x}) \longrightarrow (\mu X.M)(\bar{x}) \simeq (\mu X.M)^i(\bar{x}))$$

pertenece a H, para cualquier $i<\omega$, y operando como en el caso (i), resulta que:

$$(\mu X.M)(E_1,\ldots,E_n) \simeq (\mu X.M)^i(E_1,\ldots,E_n) \in H.$$

Como $\Delta(\mu X.M)(E_1,\ldots,E_n)$, $\Delta(\mu X.M)^i(E_1,\ldots,E_n) \in H$ entonces se tendrá $[(\mu X.M)(E_1,\ldots,E_n)] = [(\mu X.M)^i(E_1,\ldots,E_n)]$.

Pero si $\Delta((\mu X.M)^i(E_1,\ldots,E_n)) \in H$ entonces, por hipótesis de inducción, se tiene que $\mathfrak{I}((\mu X.M)^i(E_1,\ldots,E_n)) = [(\mu X.M)^i(E_1,\ldots,E_n)]$ luego:

$\mathfrak{I}((\mu X.M)(E_1,\ldots,E_n)) = \bigcup \{\mathfrak{I}((\mu X.M)^j(E_1,\ldots,E_n)) / j<\omega\}$ (por 5.4)

$= \mathfrak{I}((\mu X.M)^i(E_1,\ldots,E_n))$ (por tratarse de cpo's planos y ser $\mathfrak{I}((\mu X.M)^i(E_1,\ldots,E_n) \neq \perp_\sigma$)

$= [(\mu X.M)^i(E_1,\ldots,E_n)]$ (por hipótesis de inducción)

$= [(\mu X.M)(E_1,\ldots,E_n)]$

como queríamos demostrar. ■ Hecho 3.1

Demostración del Hecho 3.2.

Como antes, solo nos ocupamos de las nuevas construcciones. Solo es el caso, entonces, de $\neg\Delta M(E_1,\ldots,E_n) \in H$. Obsérvese que si $\neg\Delta E_i \in H$, para algún $i=1,\ldots,n$, entonces $\mathfrak{I}(E_i)=\perp_{\sigma i}$, por hipótesis de inducción, con lo que el valor de la tupla $(E_1,\ldots,E_n)$ es $\perp_\otimes$ y el de $M(E_1,\ldots,E_n)$ es $\perp_\sigma$, pues la interpretación de cualquier expresión funcional M es una función estricta. Por tanto supondremos que $\Delta E_i \in H$, para todo $i=1,\ldots,n$, con lo que $\mathfrak{I}(E_i) = [E_i]$, para todo $i=1,\ldots,n$. Entonces, distinguiendo los dos casos posibles para M, tenemos:

(i) $\neg\Delta(\lambda x_1..x_n.E)(E_1,\ldots,E_n) \in H \Rightarrow \neg\Delta E[E_1/x_1,\ldots,E_n/x_n] \in H$ ((B), pues $\Delta E_i \in H$ con lo que no es posible $\neg\Delta E_i \in H$, para ningún $i=1,\ldots,n$)

$\Rightarrow \mathfrak{I}((\lambda x_1..x_n.E)(E_1,\ldots,E_n)) = \mathfrak{I}((\lambda x_1..x_n.E))([E_1],\ldots,[E_n])$

$= \mathfrak{I}([E_1]/x_1,\ldots,[E_n]/x_n)(E)$ (por definición)

$= \mathfrak{I}(E[E_1/x_1,\ldots,E_n/x_n])$ (por el lema de sustitución)

$= \perp_\sigma$ (por hipótesis de inducción)

(11) $\neg\Delta(\mu X.M)(E_1,\ldots,E_n)\in H \Rightarrow \neg\Delta(\mu X.M)^i(E_1,\ldots,E_n)\in H$, para todo $i<\omega$ (A^*)

$\Rightarrow \overline{\mathfrak{I}}((\mu X.M)(E_1,\ldots,E_n)) = \overline{\mathfrak{I}}((\mu X.M))([E_1],\ldots,[E_n]) =$

$= \bigcup \{\overline{\mathfrak{I}}((\mu X.M)^i)([E_1],\ldots,[E_n]) \;/\; i<\omega\}$ (por 5.4)

$= \perp_\sigma$

pues $\overline{\mathfrak{I}}((\mu X.M)^i)([E_1],\ldots,[E_n]) = \perp_\sigma$, para todo $i<\omega$, por hipótesis de inducción, ya que $\neg\Delta(\mu X.M)^i(E_1,\ldots,E_n)\in H$, para todo $i<\omega$. ■ Hecho 3.2

Demostración del Hecho 3.3.

La demostración es, *mutatis mutandi*, la misma que la que se dio en §2.6.3.3, teniendo en cuenta que allí se trataban términos y aquí se tratan expresiones. ■ Hecho 3.3

Demostrado el Hecho 3.3 es evidente que la $\overline{\Sigma}$-interpretación $\overline{\mathfrak{I}}$ es modelo fuerte del conjunto de Hintikka H. ■

Si un conjunto de Hintikka aparece etiquetando nodos de una rama de un tableau entonces dicha rama no será preciso que "se prolongue más". Aquellos tableaux a los que les ocurre esto se caracterizan, como en primer orden, mediante el concepto de *tableau completo*.

4.-Definición. *Tableau completo.*

Un tableau para un conjunto de fórmulas Φ se dice completo si y solo si toda rama abierta de él es completa, esto es, el conjunto de fórmulas que etiquetan sus nodos es un conjunto H que satisface:

(1) $\Phi \subseteq H$

(2) H es un conjunto de Hintikka. ■

La completitud del método de los tableaux, que tiene el mismo sentido que para LFRP, se demuestra también mediante la construcción de un tableau canónico.

5.-Teorema. *Existencia de tableaux completos. Tableau canónico.*

Dado un conjunto de fórmulas Φ, el conjunto $\Phi_{AI} = \Phi \cup AI$, donde AI es el conjunto de axiomas de la igualdad adecuados a $\Phi \cup \{\Delta c \;/\; c\in C^\sigma,\ \sigma\in S\}$, tiene un tableau completo.

Demostración. Sea una enumeración $\Phi = \{\varphi_0, \varphi_1, \ldots, \varphi_n, \ldots\}$ de un conjunto cualquiera de $\bar{\Sigma}$-fórmulas. Consideremos una enumeración de la unión de los siguientes conjuntos (numerables) de fórmulas:

(i) $\{<\alpha, \alpha_i> / \alpha \in ALFA, 1 \leq i \leq m\}$

(ii) $\{\beta / \beta \in BETA\}$

(iii) $\{<\gamma, \gamma(E)> / \gamma \in GAMMA, E \in E^{\sigma}_{\Sigma}\}$

(iv) $\{\delta / \delta \in DELTA\}$

(v) $\{<\alpha^*, \alpha^*_i> / \alpha^* \in \omega\text{-}ALFA, i < \omega\}$

(vi) $\{\beta^* / \beta^* \in \omega\text{-}BETA\}$

Para un conjunto de $\bar{\Sigma}$-fórmulas cualquiera Ψ, definimos el siguiente concepto de *relevancia*:

(a) $<\varphi, \psi>$ es relevante para $\Psi \Leftrightarrow \varphi \in \Psi$ y $\psi \notin \Psi$

(b) β es relevante para $\Psi \Leftrightarrow \beta \in \Psi$ y $\beta_i \notin \Psi$, para ningún $i=1,..,m$

(c) δ es relevante para $\Psi \Leftrightarrow \delta \in \Psi$ y $\delta(c) \notin \Psi$, para ninguna constante auxiliar $c \in C^{\sigma}$

(d) β^* es relevante para $\Psi \Leftrightarrow \beta^* \in \Psi$ y $\beta^*_i \notin \Psi$, para ningún $i < \omega$.

El tableau completo para el conjunto de fórmulas Φ se obtiene mediante el siguiente algoritmo dado en forma de programa while:

```
begin
    s:=0;
    Σ_s := ⊙ Φ_AI ;
    while no-completo?(Σ_s) do
        begin
            Σ_{s+1} := extensión(Σ_s);
            s:= s+1
        end
end.
```

donde extensión(Σ_s) modifica Σ_s de acuerdo con el siguiente proceso:

extender toda rama abierta con la menor fórmula, o par de fórmulas, relevante al conjunto que etiqueta su hoja, de acuerdo con la regla de entre las (A)-(B^*) dadas en 7.2.

Por inducción sobre s demostremos que cada Σ_s es un tableau de ramas finitas para el conjunto de fórmulas Φ_{AI}. Para s=0, el árbol con un solo nodo etiquetado con Φ_{AI} es un tableau para Φ_{AI}, según la condición (INI) de 7.2. Supuesto que lo sea Σ_s entonces es evidente que $\Sigma_s \xrightarrow[B]{} \Sigma_{s+1}$, donde B

es el conjunto de ramas abiertas de Υ_s, por tanto Υ_{s+1} es un tableau de ramas finitas, de acuerdo con la condición (ET) de 7.2.

Por tanto, si el programa anterior se detiene en alguna etapa s, el árbol resultante es un tableau (de ramas finitas). En otro caso, obtenemos una sucesión de tableaux de ramas finitas $\langle \Upsilon_s \ / \ s\in N\rangle$ cuyo límite es, por definición, un tableau. Representemos por $\Upsilon_{\Phi AI}$ al tableau resultante en cualquiera de los dos casos, y llamémosle tableau canónico para Φ.

Veamos que $\Upsilon_{\Phi AI}$ es un tableau completo. Si $\Upsilon_{\Phi AI}$ es uno de los tableaux de ramas finitas Υ_s entonces, según se ha definido el programa que lo genera, el tableau es completo, pues solo se detiene cuando éste es el caso. Si $\Upsilon_{\Phi AI}$ es el límite de la sucesión $\langle \Upsilon_s \ / \ s\in N\rangle$ veamos que todas sus ramas abiertas -si las hay- tienen sus nodos etiquetados por conjuntos de fórmulas cuya unión forma un conjunto de Hintikka. Una vez hecho esto las ramas serán completas pues contienen a la raíz $\Phi_{AI}(\supseteq\Phi)$.

Sea pues R una rama abierta tal que Φ_R es la unión de los conjuntos de fórmulas que etiquetan sus nodos y demostremos que Φ_R es un conjunto de Hintikka. Por ejemplo, para la condición (A^*) supongamos que α^* es una fórmula ω-conjuntiva tal que $\alpha^*\in\Phi_R$ y $\alpha_1^*\notin\Phi_R$, para algún $1<\omega$. Si Φ_{Rs} es la parte de Φ_R correspondiente a restringirnos en dicha rama a Υ_s entonces, ya que $\alpha^*\in\Phi_R$, existirá $s<\omega$ tal que $\alpha^*\in\Phi_{Rs}$; pero para algún $s'>\omega$, $\langle\alpha^*, \alpha_1^*\rangle$ será el menor ítem relevante para $\Phi_{Rs'}$, y entonces α_1^* pertenecerá a $\Phi_{R(s'+1)}(\subseteq\Phi_R)$, lo que nos lleva a contradicción. De manera similar se razona con los casos restantes entre (A) y (B^*). Respecto a (E), sea θ un axioma de la igualdad adecuado a Φ_R; si θ no aparece en Φ_{AI} entonces, por definición de AI, θ no es adecuado a $\Phi\cup\{\Delta c \ / \ c\in C^\sigma, \sigma\in S\}$. Pero esto nos lleva a contradicción ya que si así fuera se daría alguna de las siguientes condiciones:

(i) θ contiene algún símbolo de Σ que no aparece en AI

(ii) θ contiene alguna constante auxiliar que no aparece en AI

(iii) θ contiene alguna variable libre que no lo está en AI.

El caso (i) es absurdo pues al ser θ adecuado a Φ_R, todos sus símbolos aparecen en el tableau y, por tanto, en la raíz, ya que en el programa anterior nunca se llegan a introducir nuevas fórmulas sino solo constituyentes de las que ya había. El caso (ii) es claramente imposible pues AI contiene todas las constantes auxiliares. Por último, el caso (iii) vuelve a ser absurdo ya que en el proceso de extensión de un tableau que no introduce nuevas fórmulas, como es este caso, nunca llegan a aparecer nuevas variables libres que no hubiera (obsérvese que la regla (C), que es quien lo

podría llegar a hacer, no lo hace por definición de expresión adecuada a un conjunto de fórmulas). En consecuencia θ aparece en Φ_{AI} y, por tanto, Φ_R contiene todos los axiomas de la igualdad adecuados a él.

Por último la condición (F) es evidente pues una rama abierta no puede contener incoherencias ya que, si así fuera, sería cerrada. ■

La demostración de la completitud del método de los tableaux es similar a la de LFP, aunque no se usa en ella el lema de König.

6.-Teorema. *Completitud del método de los tableaux.*

Si un conjunto de fórmulas Φ no es fuertemente satisfactible entonces tiene un tableau cerrado.

Demostración. Supongamos que Φ no tiene ningún tableau cerrado. Entonces su tableau canónico $\mathcal{T}_{\Phi AI}$ es abierto. Caben entonces dos posibilidades:

(i) Si $\mathcal{T}_{\Phi AI}$ es de ramas finitas entonces alguna de ellas habrá de ser abierta, pues $\mathcal{T}_{\Phi AI}$ no es cerrado.

(ii) Si $\mathcal{T}_{\Phi AI}$ no es de ramas finitas entonces alguna rama será infinita y, por tanto, abierta, ya que las ramas cerradas son finitas.

En ambos casos podemos concluir que $\mathcal{T}_{\Phi AI}$ tiene una rama abierta. Pero $\mathcal{T}_{\Phi AI}$ es completo luego toda rama abierta suya está etiquetada por un conjunto de fórmulas H que es de Hintikka y satisface $H \supseteq \Phi \cup AI$ ($\supseteq\Phi$). Como, según hemos demostrado, H es fuertemente satisfactible, entonces Φ también lo es, como queríamos demostrar. ■

10.- EL METODO DE LOS TABLEAUX Y LOS CALCULOS DE SECUENCIAS PARA LFRP.

Recordemos que los cálculos que se obtienen para la consecuencia lógica que venimos empleando, $\vdash_{sw}$, son cálculos de secuencias (cf. §2.7.1). En el capítulo anterior derivábamos secuencias de la forma (Γ, φ), donde Γ era un conjunto finito de fórmulas. Para LFRP obtendremos cálculos que derivan secuencias de la forma (Φ, φ), donde Φ es un conjunto cualquiera de fórmulas. Mantendremos la notación que escribe "$\Phi \cdot \varphi$" para referirse a (Φ, φ), así como la que utiliza comas para referirse a la unión de conjuntos de fórmulas.

La razón de que el antecedente de una secuencia pueda ser un conjunto arbitrario de fórmulas es que los cálculos que vamos a obtener producen derivaciones en las que ciertas fórmulas solo es posible concluirlas a partir de una cantidad numerable de hipótesis. Se trata de los llamados cálculos infinitarios.

1.-Definición. *Cálculos infinitarios de secuencias. Derivabilidad.*

Un cálculo infinitario de secuencias es un conjunto de esquemas de reglas de la forma:

$$\frac{\Phi_i \quad \varphi_i \ (i<\xi)}{\Phi \quad \varphi}$$

donde ξ es un ordinal menor o igual que ω.

Cuando ξ sea 0 omitiremos la parte superior de la regla y solo escribiremos la inferior.

Una secuencia $\Phi \quad \varphi$ es derivable, escrito $\Phi \vdash \varphi$, si y solo si existe una sucesión, indexada por un ordinal a lo más numerable, de secuencias que termina en ella y donde cada una se ha obtenido por aplicación de alguna regla a una cantidad (finita o numerable) de secuencias anteriores a ella en la sucesión. ■

Los conceptos de corrección y completitud de cálculos infinitarios de secuencias son similares a los finitarios (cf. §2.7.2).

2.-Definición. *Corrección y completitud de cálculos infinitarios.*

Un cálculo infinitario de secuencias se dice correcto si y solo si para todo conjunto de fórmulas $\Phi \cup \{\varphi\}$: $\Phi \vdash \varphi \Rightarrow \Phi \models \varphi$.

Un cálculo infinitario de secuencias se dice completo si y solo si para todo conjunto de fórmulas $\Phi \cup \{\varphi\}$: $\Phi \models \varphi \Rightarrow \Phi \vdash \varphi$. ■

En adelante suprimiremos el adjetivo infinitario y hablaremos simplemente de cálculos de secuencias. La relación que existe entre los tableaux y la derivabilidad en dichos cálculos es ahora la siguiente.

3.-Lema.

Si la derivabilidad en un cálculo de secuencias $\mathfrak{C}$ satisface las siguientes ocho condiciones entonces el cálculo es completo:

(0) Para todo conjunto de Σ-fórmulas Φ y todo tableau de ramas finitas $\mathcal{I}$ para Φ, si toda rama de $\mathcal{I}$ con Ψ como conjunto de fórmulas etiquetando su hoja satisface $\Psi \vdash F$ entonces $\Pi \vdash F$, siendo Π el conjunto de fórmulas de Φ usadas en la construcción de $\mathcal{I}$.

(1) $\Phi, \varphi, \neg\varphi \vdash F$

(2) $\Phi, \varphi, \neg\Delta\varphi \vdash F$

(3) $\Phi, \neg\varphi, \neg\Delta\varphi \vdash F$

(4) $\Phi, \neg\Delta x^{\sigma} \vdash F$, para toda variable x^{σ} de cualquier género σ

(5) $\Phi, \neg\Delta c^{\sigma} \vdash F$, para toda constante c^{σ} de cualquier género σ

(6) $\Phi, \neg\varphi \vdash F \Rightarrow \Phi \vdash \varphi$

(7) $\Phi \vdash \varphi \Rightarrow \Phi, \Phi' \vdash \varphi$, para todo Φ'.

Demostración. Sea $\Phi \cup \{\varphi\}$ un conjunto cualquiera de fórmulas y supongamos que $\Phi \vdash \varphi$. Entonces $\Phi \cup \{\neg\varphi\}$ no es fuertemente satisfactible. Por el teorema 9.6, $\Phi \cup \{\neg\varphi\}$ tiene un tableau cerrado de manera que todas sus ramas son finitas y están cerradas. Por las condiciones (1)-(5), todas las ramas satisfacen que sus hojas están etiquetadas por un conjunto de fórmulas Ψ de forma que $\Psi \vdash F$. Por la condición (0) se tiene que $\Pi \vdash F$, siendo Π el conjunto de fórmulas de $\Phi \cup \{\neg\varphi\}$ usadas en la construcción del tableau. Por la condición (7) resultará que $\Phi, \neg\varphi \vdash F$ luego, por la condición (6): $\Phi \vdash \varphi$, como queríamos demostrar. ■

Para expresar la condición (0) en términos de condiciones de derivabilidad más finas, demostramos el siguiente resultado.

4.-Lema.

Si la derivabilidad en un cálculo de secuencias $\mathfrak{C}$ satisface las siguientes condiciones:

(7) $\Phi \vdash \varphi \Rightarrow \Phi, \Phi' \vdash \varphi$, para todo Φ'

(8) Para toda fórmula $\alpha \in$ ALFA y todo $i=1,\ldots,m$: $\Phi, \alpha_i \vdash \varphi \Rightarrow \Phi, \alpha \vdash \varphi$

(9) Para toda fórmula $\beta \in$ BETA:

$\Phi, \beta_1 \vdash \varphi, \ldots, \Phi, \beta_m \vdash \varphi \Rightarrow \Phi, \beta \vdash \varphi$

(10) Para toda fórmula $\gamma \in$GAMMA y toda $\bar{\Sigma}$-expresión E:

$\Phi, \gamma(E) \vdash \varphi \Rightarrow \Phi, \gamma \vdash \varphi$

(11) Para toda fórmula $\delta \in$DELTA y toda constante auxiliar c que no aparezca en $\Phi \cup \{\delta, \varphi\}$:

$\Phi, \delta(c) \vdash \varphi \Rightarrow \Phi, \delta \vdash \varphi$

(12) Para toda fórmula $\alpha^{\bullet} \in \omega$-ALFA y todo $i<\omega$: $\Phi, \alpha_i^{\bullet} \vdash \varphi \Rightarrow \Phi, \alpha^{\bullet} \vdash \varphi$

(13) Para toda fórmula $\beta^{\bullet} \in \omega$-BETA: $\Phi, \beta_i^{\bullet} \vdash \varphi$ $(i<\omega) \Rightarrow \Phi, \beta^{\bullet} \vdash \varphi$

(14) Para todo axioma de la igualdad θ: $\Phi, \theta \vdash \varphi \Rightarrow \Phi \vdash \varphi$

entonces:

Para todo conjunto de Σ-fórmulas Φ, toda Σ-fórmula π y todo tableau de ramas finitas Υ para Φ, si toda rama de Υ, con Ψ como conjunto de fórmulas etiquetando su hoja, satisface $\Psi \vdash \pi$, entonces $\Pi \vdash \pi$, siendo Π el conjunto de fórmulas de Φ usadas en la construcción de Υ.

En particular, se satisface la condición (0) del lema anterior.

Demostración. Por inducción sobre la profundidad $d(\Upsilon)$ de Υ. Si $d(\Upsilon)=0$ entonces Υ está formado por un solo nodo que, según la condición (INI) de 7.2, estará etiquetado con un conjunto de fórmulas $\Phi_0 (\subseteq \Phi)$. El lema es, en este caso, obvio. Si $d(\Upsilon)>0$ distinguimos casos de acuerdo con la forma de Υ que nos dan las condiciones (i)-(vi) de 7.4.

Supongamos pues que $\Phi_R \vdash \pi$, para cada una de las ramas R de Υ, siendo Φ_R el conjunto de fórmulas que etiqueta su hoja, y veamos que $\Pi \vdash \pi$, teniendo en cuenta que, por hipótesis de inducción, esto es cierto para todo tableau de ramas finitas cuya profundidad sea menor que la de Υ. Como vimos en el lema 7.4, existe un subconjunto Ψ de Φ tal que Ψ es coherente y etiqueta la raíz de Υ. Distingamos ahora las formas posibles de Υ:

(i) Existe una fórmula conjuntiva $\alpha \in \Psi$ tal que $\alpha_i \in \Psi$, para algún $i=1,\ldots,m$, y existe un tableau de ramas finitas Υ' para un conjunto Π' que incluye la etiqueta $\Psi \cup \{\alpha_i\}$ de la raíz de Υ' de forma que se verifica Υ = Alargar(Υ', Ψ).

Por definición de Alargar, la profundidad de Υ' es menor que la de Υ. Por otra parte, $\Phi_{R'} \vdash \pi$, para toda rama R' de Υ', según se ha supuesto para Υ (recuérdese que las fórmulas pasan de padres a hijos). Por tanto, por hipótesis de inducción, $\Pi' \vdash \pi$. Ahora bien, $\Pi \supseteq \Pi' \setminus \{\alpha_i\}$ luego, por la condición (7), se tiene $\Pi, \alpha_i \vdash \pi$. Entonces, por la condición (8), se tendrá $\Pi \vdash \pi$, como queríamos demostrar.

De forma análoga se demuestra para el caso en que Υ se alargue por una fórmula ω-conjuntiva, usando la condición (12).

(ii) Existe una fórmula disyuntiva $\beta \in \Psi$ y m tableaux de ramas finitas Σ_i, cada uno para un conjunto de fórmulas Π_i que incluye la etiqueta $\Psi \cup \{\beta_i\}$ de la raíz de cada uno de ellos, de forma que se verifica Σ = **Ramificar**($\{\Sigma_i \ / \ 1 \leq i \leq m\}$, Ψ).

Como en el caso anterior, por definición de **Ramificar**, la profundidad de cada Σ_i es menor que la de Σ. Por otra parte, $\Phi_{Ri} \vdash \pi$, para toda rama Ri de Σ_i, según se ha supuesto para Σ. Por hipótesis de inducción $\Pi_i \vdash \pi$, para todo i=1,...,m. Como $\Pi \supseteq \Pi_i \setminus \{\beta_i\}$ entonces, por la condición (7), se tiene $\Pi, \beta_i \vdash \pi$, para todo i=1,...,m. Entonces, por la condición (9), tendremos que $\Pi \vdash \pi$, como queríamos demostrar.

Similarmente se demuestra cuando Σ se ramifica mediante una fórmula ω-disyuntiva, usando la condición (13). Obsérvese que en el proceso análogo al anterior es necesario usar el hecho de que el antecedente de una secuencia puede no ser finito.

(iii) Existe una fórmula universal $\gamma \in \Psi$, una expresión E adecuada a Ψ y un tableau de ramas finitas Σ' para un conjunto de fórmulas Π' que incluye la etiqueta de su raíz $\Psi \cup \{\gamma(E)\}$, de forma que Σ = **Alargar**(Σ', Ψ).

Como en los casos anteriores, Σ' tiene menos profundidad que Σ y para la etiqueta Φ_R de las hojas de todas sus ramas R se tiene $\Phi_R \vdash \pi$, por verificarse para Σ. En consecuencia, $\Pi' \vdash \pi$, por hipótesis de inducción. Como $\Pi \supseteq \Pi' \setminus \{\gamma(E)\}$ tendremos entonces que $\Pi, \gamma(E) \vdash \pi$, por la condición (7), de manera que $\Pi \vdash \pi$, por la condición (10).

(iv) Existe una fórmula existencial $\delta \in \Psi$, una constante auxiliar c que no aparece en Ψ y un tableau de ramas finitas Σ' para un conjunto de fórmulas Π' que incluye la etiqueta de su raíz $\Psi \cup \{\delta(c)\}$, de forma que Σ = **Alargar**(Σ', Ψ).

De la misma forma que antes resulta que $\Pi, \delta(c) \vdash \pi$. Como c no aparece en Ψ y $\delta \in \Psi$ entonces c no aparece en δ. Como $\Pi \cup \{\pi\}$ es un conjunto de Σ-fórmulas, tampoco aparece en $\Pi \cup \{\pi\}$. Por la condición (11), $\Pi \vdash \pi$, como buscábamos.

(v) Existe una fórmula $\varphi \in \Phi$ y un tableau de ramas finitas Σ' para un conjunto de fórmulas Π' que incluye la etiqueta de su raíz $\Psi \cup \{\varphi\}$, de forma que Σ = **Alargar**(Σ', Ψ).

Como para los casos anteriores resulta que $\Pi' \vdash \pi$, por hipótesis de inducción. Como $\varphi \in \Pi$ resultará que $\Pi' = \Pi$, luego $\Pi \vdash \pi$.

(vi) Existe un axioma de la igualdad θ adecuado a Ψ y un tableau de ramas finitas Σ' para un conjunto de fórmulas Π' que incluye la etiqueta de su raíz $\Psi \cup \{\theta\}$, de forma que Σ = **Alargar**(Σ', Ψ).

Como en los caso (i)-(iv) resulta que $\Pi, \theta \vdash \pi$ de manera que, por la condición (14), se tendrá $\Pi \vdash \pi$, como queríamos demostrar. ■

Por tanto podemos demostrar la completitud de un cálculo de secuencias, sin más que aplicar los dos lemas anteriores.

5.-Teorema. *Condiciones de completitud.*

Si la derivabilidad en un cálculo de secuencias $\mathfrak{C}$ satisface las condiciones (1)-(14) de los lemas 3 y 4 entonces $\mathfrak{C}$ es completo. ■

11.- UN CALCULO DE SECUENCIAS PARA LFRP: $\mathfrak{C}\mathfrak{Z}\mathfrak{R}\mathfrak{P}$.

De la condición (13) del lema 3 de la sección anterior observamos que los cálculos de secuencias que obtendremos para LFRP son infinitarios. En efecto, para poder deducir que una función recursiva no está definida en un argumento dado habremos de demostrar que no lo está en ninguna de sus aproximaciones y tener en cuenta que, según 5.4, la unión de éstas define plenamente a la primera. Esta será la característica esencial del cálculo que definamos en esta sección; un cálculo, que en lo demás, seguirá las pautas marcadas por el de primer orden para LFP.

Como hicimos, pues, en el capítulo anterior, consideremos el siguiente cálculo de secuencias, al que denominaremos $\mathfrak{C}\mathfrak{Z}\mathfrak{R}\mathfrak{P}$, definido por el conjunto de esquemas de reglas dado por:

REGLAS ESTRUCTURALES

(IHip) $\dfrac{\Phi \quad \varphi}{\Phi, \Phi' \quad \varphi}$

IGUALDAD

(REF) $E \simeq E$

(IF) $(\chi \rightarrow E) \simeq E$

(IFT) χ $(\chi \rightarrow E, E') \simeq E$

(IFE) $\neg\chi$ $(\chi \rightarrow E, E') \simeq E'$

(ABS) $(\lambda x_1 .. x_n . E)(E_1, .., E_n) \simeq E[E_1/x_1, \ldots, E_n/x_n]$

(REC) $(\mu X.M)(E_1, \ldots, E_n) \simeq (\mu X.M)^i(E_1, \ldots, E_n) \quad (i<\omega)$

(Subs) $\dfrac{\Phi \quad \varphi[E/x]}{\Phi, E \simeq E' \quad \varphi[E'/x]}$

(IIg) $\dfrac{\Phi \ \neg\Delta E_i}{\Phi \ E_1 \simeq E_2}$ $\qquad$ $\dfrac{\Phi \ \neg\Delta E_i}{\Phi \ \neg(E_1 \simeq E_2)}$ $(1 \leq i \leq 2)$

REGLAS PARA CONECTIVAS

(IEPr) $\dfrac{\Phi \ \neg\Delta E_i}{\Phi \ R(E_1,\ldots,E_n)}$ $\qquad$ $\dfrac{\Phi \ \neg\Delta E_i}{\Phi \ \neg R(E_1,\ldots,E_n)}$ $(1 \leq i \leq n)$

(I$\wedge$C) $\dfrac{\Phi \ \neg\varphi \ , \ \Phi \ \neg\psi}{\Phi \ \neg(\varphi \vee \psi)}$ $\qquad$ (I$\vee$C) $\dfrac{\Phi \ \varphi}{\Phi \ \varphi \vee \psi}$ $\qquad$ $\dfrac{\Phi \ \varphi}{\Phi \ \psi \vee \varphi}$

(I$\neg\neg$C) $\dfrac{\Phi \ \varphi}{\Phi \ \neg\neg\varphi}$

(Ctp$_1$) $\dfrac{\Phi, \ \varphi \ \psi}{\Phi, \ \neg\psi \ \neg\varphi}$ $\qquad$ (Ctp$_2$) $\dfrac{\Phi, \ \neg\varphi \ \neg\psi}{\Phi, \ \psi \ \varphi}$

(RFT) $\dfrac{\Phi, \ \neg\varphi \ F}{\Phi \ \varphi}$ $\qquad$ (MP$_\Delta$) $\dfrac{\Phi \ \varphi\rightarrow\psi, \ \Phi \ \varphi, \ \Phi \ \Delta\varphi}{\Phi \ \psi}$

(E$\rightarrow$C) $\dfrac{\Phi \ \varphi\rightarrow\psi}{\Phi, \ \varphi \ \psi}$

(Ctd) $\varphi, \ \neg\varphi \ \psi$

$\varphi, \ \neg\Delta\varphi \ \psi$

$\neg\varphi, \ \neg\Delta\varphi \ \psi$

$\neg\Delta x^\sigma \ \psi$ para toda variable $x^\sigma \in V^\sigma$ de cualquier género σ

$\neg\Delta c^\sigma \ \psi$ para toda constante auxiliar $c^\sigma \in C^\sigma$ de cualquier género σ

REGLAS PARA CUANTIFICADORES

(I$\exists$C) $\dfrac{\Phi \ \Delta E \wedge \varphi[E/x^\sigma]}{\Phi \ \exists x^\sigma \varphi}$ donde $E \in E^\sigma_\Sigma$ $\qquad$ (I$\forall$C) $\dfrac{\Phi \ \neg\varphi[c^\sigma/x^\sigma]}{\Phi \ \neg\exists x^\sigma \varphi}$ donde $c^\sigma \in C^\sigma$ y no aparece en $\Phi \cup \{\varphi\}$

REGLAS SOBRE DEFINICION

(IFun) $$\frac{\Phi\ \neg\Delta E_i}{\Phi\ \neg\Delta f(E_1,\ldots,E_n)}$$ (IVarFun) $$\frac{\Phi\ \neg\Delta E_i}{\Phi\ \neg\Delta X(E_1,\ldots,E_n)}\ (1\le i\le n)$$

(IIF) $$\frac{\Phi\ \neg\chi}{\Phi\ \neg\Delta(\chi\rightarrow E)} \qquad \frac{\Phi\ \neg\Delta E}{\Phi\ \neg\Delta(\chi\rightarrow E)}$$

(DIF) $$\frac{\Phi\ \chi,\ \Phi\ \Delta\chi,\ \Phi\ \Delta E}{\Phi\ \Delta(\chi\rightarrow E)}$$ (DIFT) $$\frac{\Phi\ \Delta\chi,\ \Phi\ \chi\rightarrow\Delta E,\ \Phi\ \neg\chi\rightarrow\Delta E'}{\Phi\ \Delta(\chi\rightarrow E,\ E')}$$

(IIFT) $$\frac{\Phi\ \chi\rightarrow\neg\Delta E,\ \Phi\ \neg\chi\rightarrow\neg\Delta E'}{\Phi\ \neg\Delta(\chi\rightarrow E,\ E')}$$

(IPr) $$\frac{\Phi\ R(E_1,\ldots,E_n),\ \Phi\ \neg R(E_1,\ldots,E_n)}{\Phi\ \neg\Delta R(E_1,\ldots,E_n)}$$

(DAbs) $$\frac{\Phi\ \Delta E[E_1/x_1,\ldots,E_n/x_n],\ \Phi\ \Delta E_1,\ \ldots,\ \Phi\ \Delta E_n}{\Phi\ \Delta(\lambda x_1..x_n.E)(E_1,\ldots,E_n)}$$

(IAbs) $$\frac{\Phi\ \neg\Delta E[E_1/x_1,\ldots,E_n/x_n]}{\Phi\ \neg\Delta(\lambda x_1..x_n.E)(E_1,\ldots,E_n)} \qquad \frac{\Phi\ \neg\Delta E_i}{\Phi\ \neg\Delta(\lambda x_1..x_n E)(E_1,\ldots,E_n)}\ (1\le i\le n)$$

(DRec) $$\frac{\Phi\ \Delta(\mu X.M)^i(E_1,\ldots,E_n)}{\Phi\ \Delta(\mu X.M)(E_1,\ldots,E_n)}\ (i<\omega)$$

(IRec) $$\frac{\Phi\ \neg\Delta(\mu X.M)^i(E_1,\ldots,E_n)\ \ (i<\omega)}{\Phi\ \neg\Delta(\mu X.M)(E_1,\ldots,E_n)}$$

(DE) $\Delta\Delta E$ para cualquier expresión E

Gran parte de las reglas aparecían ya en el cálculo $\mathfrak{CJP}$ que vimos en el capítulo anterior. Respecto de las nuevas hacemos notar que seguimos nombrándolas nemotécnicamente; así por ejemplo, (DRec) se refiere a la Definición de la Recursión.

Obsérvese que la regla (IRec) es infinitaria y permite concluir que una función, definida por recursión, no está definida en unos argumentos si no lo está en ninguna de sus aproximaciones.

Es fácil demostrar que los esquemas de reglas del cálculo $\mathfrak{GP}$ que se referían a la definición de fórmulas (cf. en §2.8 los llamados (DIg), (D¬), (Dv), (D∃)) son derivables en $\mathfrak{GPR}$, sin más que tener en cuenta cómo se definió la aplicación de Δ a una fórmula cualquiera en términos de la aplicación a predicados (cf. 4.1). En efecto, para (DIg) y (Dv) usar (I¬¬C), el resto de los casos son inmediatos.

Demostrado esto es fácil ver que todas las reglas derivadas en §2.9 usadas para probar la completitud de $\mathfrak{GP}$, son también derivadas en nuestro cálculo $\mathfrak{GPR}$, en particular los esquemas (Hip), (E¬¬C) e (I→C).

Para terminar esta sección veamos un ejemplo de derivación en $\mathfrak{GPR}$. Se trata de formalizar mediante nuestro cálculo la refutación que obtuvimos, en forma de tableau cerrado, al final de la sección 7. Allí demostramos que se tenía $\neg(1\simeq 0) \vdash \neg\Delta(\mu X.M)(1)$, donde M era la expresión funcional $\lambda y.\, y\simeq 0 \rightarrow 1, (y\simeq 1 \rightarrow X(1),\ X(y-2))$; recordar que abreviábamos el término $\mathrm{suc}(\mathrm{suc}(\overset{n}{\ldots}(\mathrm{suc}(0))\ldots))$ mediante n. Demostremos entonces que la secuencia $\neg(1\simeq 0)\ \ \neg\Delta(\mu X.M)(1)$ es derivable. Para ello hacemos:

(1) $\neg(1\simeq 0) \vdash \neg\Delta(\mu X.M)^0(1)$ (recuérdese que $(\mu X.M)^0$ representa Ω)

1.	$1\simeq 1$	(REF)
2.	$\neg\neg(1\simeq 1)$	(I¬¬C), 1
3.	$\neg(1\simeq 0)\ \ \neg\neg(1\simeq 1)$	(IHip), 2
4.	$\neg(1\simeq 0)\ \ \neg\Delta(\neg(1\simeq 1)\rightarrow 1)$	(IIF), 3
5.	$\neg(1\simeq 0)\ \ \neg\Delta(\lambda x.\neg(x\simeq x)\rightarrow x)(1)$	(IAbs), 4
6.	$\neg(1\simeq 0)\ \ \neg\Delta(\mu X.M)^0(1)$	Def. de Ω, 5

(2) $\neg(1\simeq 0) \vdash \neg\Delta(\mu X.M)^1(1) \Rightarrow \neg(1\simeq 0) \vdash \neg\Delta(\mu X.M)^{1+1}(1)$

1.	$\neg(1\simeq 0)\ \ \neg\Delta(\mu X.M)^1(1)$	Hipótesis
2.	$\neg(1\simeq 0)\ \ \neg(1\simeq 1)\vee\neg\Delta(\mu X.M)^1(1)$	(IvC), 1
3.	$1\simeq 1$	(REF)
4.	$\neg(1\simeq 0)\ \ 1\simeq 1$	(Hip), 3
5.	$\neg(1\simeq 0)\ \ \neg\neg(1\simeq 1)$	(I¬¬C), 4
6.	$\neg(1\simeq 0)\ \ \neg\neg(1\simeq 1)\vee\neg\Delta(\mu X.M)^1(1-2)$	(IvC), 5
7.	$\neg(1\simeq 0)\ \ \neg\Delta(1\simeq 1\rightarrow(\mu X.M)^1(1),\ (\mu X.M)^1(1-2))$	(IIFT), 2, 6
8.	$\neg(1\simeq 0)\ \ \neg\neg(1\simeq 0)\vee\neg\Delta(1\simeq 1\rightarrow(\mu X.M)^1(1),\ (\mu X.M)^1(1-2))$	(IvC), 7
9.	$\neg(1\simeq 0)\ \ \neg(1\simeq 0)$	(Hip)
10.	$\neg(1\simeq 0)\ \ \neg(1\simeq 0)\vee\neg\Delta 1$	(IvC), 9
11.	$\neg(1\simeq 0)\ \ \neg\Delta(1\simeq 0\rightarrow 1, (1\simeq 1\rightarrow(\mu X.M)^1(1),\ (\mu X.M)^1(1-2)))$	(IIFT), 10, 8
12.	$\neg(1\simeq 0)\ \ \neg\Delta(\lambda y.\, y\simeq 0\rightarrow 1, (y\simeq 1\rightarrow(\mu X.M)^1(1),\ (\mu X.M)^1(y-2)))(1)$	(IAbs), 11
13.	$\neg(1\simeq 0)\ \ \neg\Delta(\mu X.M)^{1+1}(1)$	Def. $(\mu X.M)^{1+1}$, 12

Con (1) y (2) probamos, por inducción sobre i, que $\neg(1\approx 0) \vdash \neg\Delta(\mu X.M)^i(1)$, para todo $i<\omega$. Con la regla (IRec) resulta entonces que $\neg(1\approx 0) \vdash \neg\Delta(\mu X.M)(1)$, como queríamos demostrar.

12.- CORRECCION DEL CALCULO DE SECUENCIAS $\mathfrak{GJP}$.

La corrección del cálculo de secuencias que acabamos de definir se establece a través de la corrección de los esquemas de reglas que lo forman, concepto este último que fue introducido en el capítulo anterior. Recordemos que por un esquema de regla correcto entendíamos aquel que preservaba la consecuencia lógica, esto es, si las premisas son secuencias correctas (es decir, el consecuente es consecuencia lógica del antecedente) entonces también lo es la conclusión.

1.-Lema.

Todos los esquemas de reglas del cálculo $\mathfrak{GJP}$ son correctos.

Demostración. Como vimos en §2.10.2, si un esquema de regla es de la forma:

$$\frac{\Phi\ \ \neg\alpha_i}{\Phi\ \ \neg\alpha}\ \ (1\leq i\leq m)$$

para alguna fórmula $\alpha\in$ALFA y constituyente suyo α_i, $i=1,..,m$, entonces es correcto (cf. allí la demostración).

Según esto, son correctas, salvo doble negación, las reglas (IIg), (IEPr), (IvC), (IFun), (IVarFun), (IIF) e (IAbs).

Análogamente, un esquema de regla de la forma:

$$\frac{\Phi\ \ \neg\beta_1,\ldots,\ \ \Phi\ \ \neg\beta_m}{\Phi\ \ \neg\beta}$$

para alguna fórmula $\beta\in$BETA y constituyentes suyos $\beta_1,\ldots,\beta_m$, es correcto.

Por tanto son correctas salvo, de nuevo, introducción o eliminación de dobles negaciones, (I$\wedge$C), (DIFT), (IIFT), (IPr) y (DAbs).

Por otra parte, las demostraciones de la corrección de (DIF), (I$\exists$C), (IvC), (I$\neg\neg$C), (Ctp$_1$), i=1, 2, (RFT), (Ctd), (E$\rightarrow$C), (MP$_\Delta$), (IHip), (REF), (IF), (IFT), (IFE) y (Subs) son análogas a las vistas para $\mathfrak{GJP}$.

Para la demostración de la corrección de las reglas (DRec) e (IRec) usamos las condiciones ω-ALFA y ω-BETA que vimos en 6.2. Por último. la de (ABS) y (REC) se sigue del hecho de que los axiomas:

$$\forall y_1..\forall y_n(\Delta E[y_1/x_1,\ldots,y_n/x_n] \longrightarrow (\lambda x_1..x_n.E)(y_1,\ldots,y_n)\approx E[y_1/x_1,\ldots,y_n/x_n])$$

y:

$$\forall\vec{x}(\Delta(\mu X.M)^i(\vec{x}) \longrightarrow (\mu X.M)(\vec{x}) \approx (\mu X.M)^i(\vec{x})) \quad (i<\omega)$$

son ciertos en toda interpretación. Entonces, de la semántica del lenguaje es fácil deducir que el consecuente no es falso para ninguna tupla de argumentos, como expresan los respectivos axiomas del cálculo. ■

La corrección del cálculo $\mathfrak{CJR}\mathfrak{P}$ se sigue ahora de forma inmediata haciendo inducción transfinita sobre la derivación. Para ello basta asociar un ordinal a cada derivación y ello se hace de forma obvia. Cada secuencia en una derivación ocupa el *orden* que determina el supremo de los *sucesores de los órdenes* de las secuencias anteriores a ella en la derivación.

2.-Teorema. *Corrección de* $\mathfrak{CJR}\mathfrak{P}$.

Sea $\Phi \cup \{\varphi\}$ un conjunto cualquiera de fórmulas entonces:

$$\Phi \vdash \varphi \Rightarrow \Phi \models \varphi. \quad ■$$

13.- COMPLETITUD DEL CALCULO DE SECUENCIAS $\mathfrak{CJR}\mathfrak{P}$.

La completitud de $\mathfrak{CJR}\mathfrak{P}$ se demuestra usando las quince condiciones que se obtuvieron en el teorema 10.5. Para probar las que se refieren a las fórmulas ALFA, BETA, ω-ALFA y ω-BETA vemos un resultado previo, análogo al que ya vimos en el capítulo anterior (cf. §2.11.1).

1.-Lema.

Dado un conjunto de fórmulas cualquiera Φ:

(a) Para toda fórmula $\alpha\in$ALFA y todo constituyente α_i, $1\le i\le m$, se tiene: $\Phi \vdash \neg\alpha_i \Rightarrow \Phi \vdash \neg\alpha$.

(b) Para toda fórmula $\beta\in$BETA con constituyentes β_i, $1\le i\le m$:

$$\Phi \vdash \neg\beta_1,\ldots,\ \Phi \vdash \neg\beta_m \Rightarrow \Phi \vdash \neg\beta$$

(c) Para toda fórmula $\alpha^\bullet\in\omega$-ALFA y todo constituyente $\alpha_i^\bullet$, $1<\omega$, se tiene: $\Phi \vdash \neg\alpha_i^\bullet \Rightarrow \Phi \vdash \neg\alpha^\bullet$.

(d) Para toda fórmula $\beta^\bullet\in\omega$-BETA con constituyentes $\beta_i^\bullet$, $1<\omega$:

$$\Phi \vdash \neg\beta_i^\bullet\ (1<\omega) \Rightarrow \Phi \vdash \neg\beta^\bullet$$

Demostración. (a) El esquema de regla que establece (α1) es (IEPr) y el de (α2), también, usando (I$\neg\neg$C). Lo mismo cabe decir para (α3) y (α4) y el

esquema (IIg). Para (α5) se tiene (I¬¬C), (IFun) para (α7), (IVarFun) para (α8), (IAbs) para (α9) e (IIF) para (α10). Usando (I¬¬C) y (E¬¬C) se prueba para (α6), con (IvC).

(b) Razonando de la misma manera que en el apartado anterior se tiene que (IʌC) establece (β1), (IIFT) e (IPr) establecen (β2) y (β4), respectivamente, y (DAbs) establece (β6), usando (I¬¬C) y (E¬¬C) para las dos últimas. Para (β3) y (β5) se pueden usar las mismas derivaciones que se vieron en §2.11.1.(b).

(c) y (d) son inmediatamente demostrados con las reglas (DRec) e (IRec), respectivamente, usando (I¬¬C) y (E¬¬C), para el primero. ■

La completitud del cálculo se obtiene demostrando que satisface las quince condiciones del teorema 10.5.

2.-Teorema. *Completitud de* $\mathfrak{CJRP}$.

El cálculo de secuencias $\mathfrak{CJRP}$ es un cálculo completo para LFRP.

Demostración. Las condiciones (1)-(5) del teorema 10.5 son obviamente satisfechas usando los esquemas de reglas (Ctd) e (IHip). La condición (6) es el esquema de regla (RFT) y la (7), (IHip). Para la condición (8) razonamos como sigue:

$\Phi, \alpha_1 \vdash \varphi$ (por hipótesis) $\Rightarrow$

$\Rightarrow \Phi, \neg\varphi \vdash \neg\alpha_1$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Phi, \neg\varphi \vdash \neg\alpha$ (por lema 1.(a)) $\Rightarrow$

$\Rightarrow \Phi, \alpha \vdash \varphi$ (por (Ctp_2))

Para las condiciones (9), (10) y (11) sirven las demostraciones dadas en §2.11.2 para sus análogas, entendiendo que para las dos últimas se particulariza y se generaliza sobre expresiones en lugar de sobre términos.

La condición (12) se obtiene como la (8), a partir del lema 1.(c) mientras que la (13), como la (9), a partir del lema 1.(d). Así, para esta última, se tiene:

$\Phi, \beta_1^\bullet \vdash \varphi \quad (1<\omega)$ (por hipótesis) $\Rightarrow$

$\Rightarrow \Phi, \neg\varphi \vdash \neg\beta_1^\bullet \quad (1<\omega)$ (por (Ctp_1)) $\Rightarrow$

$\Rightarrow \Phi, \neg\varphi \vdash \neg\beta^\bullet$ (por lema 1.(d)) $\Rightarrow$

$\Rightarrow \Phi, \beta^\bullet \vdash \varphi$ (por (Ctp_2))

Respecto de la condición (14) cabe decir lo mismo que para su análoga del capítulo anterior, es decir, por hipótesis se tiene que $\Phi, \theta \vdash \varphi$

luego $\Phi \vdash \theta \rightarrow \varphi$, por $(I \rightarrow C)$; si demostramos que $\Phi \vdash \theta$ y que $\Phi \vdash \Delta\theta$, bastará aplicar la regla (MP_{Δ}) para concluir que $\Phi \vdash \varphi$.

Como hicimos en el capítulo anterior, más en general demostramos, en el tercer apéndice de este trabajo, que $\vdash \theta$ y $\vdash \Delta\theta$ se verifica para cualquier axioma de la igualdad θ. ■

Como en el capítulo anterior, terminamos haciendo observar que la lógica satisface el teorema de Löwenheim-Skolem, siendo la demostración, similar a la que dimos entonces. No podemos decir lo mismo sobre la compacidad, ya que la prueba de una secuencia puede hacer uso, como hemos visto, de una cantidad no finita de hipótesis.

14.- TRABAJOS RELACIONADOS.

El estudio de aspectos relacionados con las lógicas de programas, tratados desde un punto de vista puramente funcional, aparece en los trabajos de Goerdt [Goe 85] [Goe 86], donde el λ-cáluclo es usado para demostrar la completitud relativa de un cálculo de Hoare para el lenguaje imperativo L_4 (cf. [Cla 79] y [Apt 81], para una introducción al tema). Sin embargo, los sistemas presentados allí solo se ocupan de la corrección parcial, mientras que la lógica LFRP abarca también la total.

El tratamiento de las funciones parcialmente definidas forma también parte actualmente de varias áreas de investigación sobre aspectos téoricos de la informática, diferentes de aquella que se refiere al razonamiento formal sobre los programas, como citábamos en la introducción.

Cabe citar, entre otras, las dos siguientes. La primera se refiere a los estudios sobre especificación formal de tipos abstractos de datos [BW 82] [Pro 85] [Loe 87]. Este último centra la discusión en especificaciones permitiendo funciones parciales y presenta una lógica para álgebras estrictas.

La segunda se refiere a la programación lógica y destacamos los trabajos siguientes. Fitting en [Fit 85] [Fit 86] presenta tres extensiones de la semántica de punto fijo de PROLOG: saturada_P, fuerte_P y débil_P, para programas formados por conjuntos P de definiciones $R(\bar{x}) \leftarrow \varphi(\bar{x})$, donde φ es una Σ-fórmula y R es un símbolo de predicado (no se permite la igualdad) que no ocurre dos veces en P. Para hacerlo introduce el concepto de Σ-interpretación parcial $\mathfrak{I}$ que no es más que un conjunto consistente de

Σ-fórmulas atómicas (sin igualdad), con signo asociado T y F, que expresa si la fórmula es cierta o falsa en J, de forma que aquella que no pertenezca a J está indefinida. Las tres semánticas coinciden con la convencional sobre programas con claúsulas de Horn puras. [Fit 86] da además, para la semántica débil$_P$, un procedimiento de prueba correcto y completo basado en el método de los tableaux.

Sobre sistemas trivalorados en el campo de la programación lógica son también las investigaciones de Delahaye. En [Del 88] se define el concepto de regla trivalorada en la que se permite el uso de la negación tanto en la cabeza como en el cuerpo. Un conjunto de tales reglas es un programa, probándose que se satisface la propiedad clásica de que un tal programa tiene un más pequeño modelo, trivalorado. La aplicación de estos estudios para la obtención de intérpretes trivalorados para PROLOG son tratados en [Del 88] y [DM 89].

CAPITULO 4
APLICACIONES

La semántica de un lenguaje de programación puede ser definida desde dos puntos de vista: operacional y denotacional. Según el primero, un programa S se entiende como una construcción que, fijados unos valores de sus variables globales (libres), los transforma dando lugar, si el programa no diverge, a unos valores de salida. A su vez, la forma en que hace esto se define recursivamente sobre la estructura de S, es decir, sobre la forma en que se ha construido. Con otras palabras, la semántica de S se define en función de cómo *opera* S.

Desde el punto de vista denotacional, el significado de un programa es una función, pero con un enfoque diferente. Mientras que operacionalmente la función se entiende en su sentido intensional, centrando el interés en la forma en que se obtiene la salida a partir de los valores de entrada, denotacionalmente la función se entiende en su sentido extensional, considerando el grafo que *denota*.

El λ-cálculo es un formalismo adecuado para definir funciones, de manera que resultaría apropiado para tratar en él la semántica denotacional. Ello ha sido así y es conocida la relación que existe entre uno y otra (cf. por ejemplo, [Sco 76] [Sco 82]).

Como vimos en el capítulo anterior, la lógica para funciones recursivas parciales LFRP es un λ-cálculo, que dispone además de un operador μ de recursión; por tanto es natural plantearse la posibilidad de usarlo para describir el significado de lenguajes de programación.

Los lenguajes de programación imperativos disponen de construcciones difíciles de expresar operacionalmente. Así, los mecanismos de paso de parámetros por variable (que incluyen la compartición de memoria entre variables) o las declaraciones de procedimientos con variables globales (con lo que supone de efectos colaterales en las llamadas a los mismos) requieren procesos sintácticos engorrosos.

Nuestro propósito es trasladar estos problemas desde un plano sintáctico a uno semántico y definir los procesos, que en él llevemos, mediante expresiones de nuestro formalismo funcional. Hecho esto así, ciertos aspectos de las lógicas de programas de lenguajes imperativos podrán ser explicados

con mayor claridad y simplicidad.

Iniciamos el capítulo presentando la sintaxis de un lenguaje de programación imperativo *LI* de tipo ALGOL. Se caracteriza, esencialmente, por la posibilidad de realizar declaraciones locales de variables y procedimientos; estos últimos disponen de dos mecanismos de paso de parámetros: por valor y por referencia. El resto de las construcciones son las habituales: asignación, composición, bifurcación condicional, llamadas a procedimientos. Respecto a éstas se permite que puedan ser a procedimientos no declarados.

La semántica de un lenguaje como *LI* es difícil de expresar de una forma operacional, debido fundamentalmente al paso de parámetros. Nuestro propósito en la sección segunda es estudiarla distinguiendo de forma precisa entre los datos que maneja un programa, que vendrán dados por los clásicos estados de memoria, y las direcciones, que vendrán dadas por aplicaciones entre variables que indiquen dónde se ubican éstas. Con estos dos conceptos puede ser expresada la semántica del lenguaje, teniendo en cuenta que la de una declaración local de variable o procedimiento utiliza variables "nuevas", evitando que la variable declarada (o, en su caso, los parámetros formales usados en la declaración) quede afectada colateralmente.

Todas estas operaciones, que se describen en un plano semántico, es posible llevarlas a cabo sintácticamente dentro de LFRP. Ello hacemos en la secciones tercera y cuarta, donde se muestra qué géneros y qué símbolos sobre ellos son necesarios para hacerlo. Una signatura que disponga de esto permitirá construir expresiones funcionales que, interpretadas de forma apropiada, simulen el comportamiento de los programas.

La semántica de un programa vendrá dada entonces a través de dos expresiones funcionales. En una se explica el comportamiento de la ejecución de un programa a partir de un estado de memoria, una ubicación de sus direcciones y un contador que nos diga la primera dirección no usada hasta entonces; en la otra, la semántica propiamente dicha de un programa S es expresada por la transformación de estados que produce la ejecución de S cuando inicialmente no se ha reubicado ninguna variable y el contador es aquel que señala a la primera dirección (variable) que no aparece en S.

Ambos comportamientos, ejecución y significado, se obtienen a través de expresiones funcionales de tipo apropiado, de forma que empleamos LFRP como herramienta para especificar denotacionalmente la semántica de los programas de *LI*.

Por último dedicamos la sección quinta a mostrar cómo es posible traducir una lógica dinámica, basada en la clase de los programas de *LI*, a expresiones de nuestro formalismo LFRP, sobre la signatura que hemos introducido en las secciones anteriores. Para ello suponemos que una fórmula "diamante" del tipo $\langle S \rangle \varphi$, se hace falsa siempre que el programa S diverja, y hacemos uso, para llevar a cabo la traducción, del operador de definición Δ aplicado a la expresión cuya interpretación es el significado de S. La traducción, a su vez, preserva el significado de las fórmulas de tal manera que el valor booleano (de los tres posibles) es el mismo para una fórmula y su traducida.

1.- SINTAXIS DE UN LENGUAJE DE PROGRAMACION IMPERATIVO PARA PROGRAMAS RECURSIVOS.

El lenguaje que aquí presentamos es una ligera variante del lenguaje de programas recursivos de [MM 83]. Se trata de un lenguaje de programación imperativo cuya característica esencial es la declaración de procedimientos (eventualmente, mutuamente) recursivos y la posibilidad de llamadas a procedimientos no declarados (generalmente llamados globales). En ambos casos, las llamadas pasan una serie de parámetros por valor y otros por referencia.

Sobre la clase de los programas así definidos se construye una lógica dinámica de programas, al estilo de las usuales sobre otra clase de programas (cf. por ejemplo, entre otras, QDL, sobre la clase de los programas regulares, o la de los while-programas en [Har 79] [Har 84]).

Para definir el lenguaje supondremos dada, pues, una signatura de primer orden Σ a la manera de la lógica clásica, es decir, Σ está compuesta por un conjunto de símbolos de función $\{f/n, f'/n', \ldots\}$, cada uno con una aridad asociada n, n', .. (entendiendo, como siempre, que aquellos con aridad cero representan constantes c) y un conjunto de símbolos de predicado $\{R/n, R'/n', \ldots\}$, con una aridad asociada n, n', ...

Supondremos también dados unos conjuntos numerables de variables (de individuo) $V=\{x_i \mid i<\omega\}$ y de variables de procedimiento $PV=\{X_i/m\triangleright n \mid i<\omega\}$. Las primeras hacen el papel de variables de un programa y las segundas el de identificadores de procedimientos; estas últimas llevan asociada una aridad $m\triangleright n$ que indica su número de parámetros por valor (m) y por referencia (n). Para que la notación sea más sencilla emplearemos las letras P, Q, ..., para referirnos a los elementos de PV, suprimiendo, siempre que no haya ambigüedad, la aridad asociada.

Bajo estos supuestos, la sintaxis del lenguaje de programación que presentamos es la siguiente.

1.-Definición. *Sintaxis.*

El conjunto P_Σ de los programas recursivos S sobre Σ, y F_Σ, de las fórmulas φ sobre Σ, se definen mediante las siguientes reglas en forma de Backus-Naur:

S::= skip |
div |

$x := t$ ($x \in V$ y $t \in T_\Sigma$) |

S; S' |

if φ then S else S' fi ($\varphi \in F_\Sigma$, de primer orden y libre de cuantificadores) |

dec $x=t$ do S od ($x \in V$ y $t \in T_\Sigma$) |

proc $P(\bar{x} \triangleright \bar{y})$:S do S' od ($P/m \triangleright n \in PV$, $\bar{x}$ representa una sucesión de m variables diferentes $x_1,\ldots,x_m$ e $\bar{y}$, otra disjunta de n variables, también diferentes, $y_1,\ldots,y_n$, donde x_i, $y_j \in V$, $i=1,\ldots,m$, $j=1,\ldots,n$) |

$P(\bar{t} \triangleright \bar{y})$ ($P/m \triangleright n \in PV$, $\bar{t}$ representa una sucesión de m Σ-términos $t_1,\ldots,t_m$ e $\bar{y}$, una de n variables $y_1,\ldots,y_n$, $t_i \in T_\Sigma$, $i=1,\ldots,m$, $y_j \in V$, $j=1,\ldots,n$)

$\varphi ::=$ $t=t'$ (t, $t' \in T_\Sigma$) |

$R(t_1,\ldots,t_n)$ (R símbolo n-ario de relación de Σ, $t_i \in T_\Sigma$, $i=1,\ldots,n$) |

$\neg\varphi$ |

$\varphi \vee \psi$ |

$\exists x \varphi$ ($x \in V$) |

$\langle S \rangle \varphi$

■

Como puede verse, la clase de programas que acabamos de definir contiene las construcciones habituales de un lenguaje de programación tipo ALGOL, esto es, asignación, composición, instrucción condicional, declaración de variables locales, declaración local de procedimientos y llamadas a éstos. Así mismo, añadimos dos construcciones especiales: skip, un programa que no hace nada, y div, un programa que diverge siempre.

En un programa como proc $P(\bar{x} \triangleright \bar{y})$:S do S' od decimos que $P(\bar{x} \triangleright \bar{y})$:S es la declaración del procedimiento P, que S es el cuerpo de la declaración y que $\bar{x}$ (resp. $\bar{y}$) es la lista de *parámetros formales por valor* (resp. *por referencia*) de la declaración de P. Entonces en una llamada de la forma $P(\bar{t} \triangleright \bar{y})$ diremos que $\bar{t}$ (resp. $\bar{y}$) es la lista de *parámetros actuales* que se pasan *por valor* (resp. *por referencia*).

Con la clase de programas que acabamos de definir es posible expresar otras construcciones usuales en un lenguaje de programación imperativo. Veamos algunas de ellas.

Con **skip** es posible simular la construcción usual **if** φ **then** S **fi** -que converge cuando φ es falsa- mediante **if** φ **then** S **else skip fi**, y con **div**, la construcción **if** φ **then** S **fi** -que diverge cuando φ es falsa- mediante **if** φ **then** S **else div fi**. Así mismo, el test φ?, cuyo comportamiento deseado es continuar la ejecución si se cumple φ y diverger, en otro caso, puede simularse mediante **if** φ **then skip else div fi**.

La construcción **while** φ **do** S **od** puede expresarse mediante el siguiente programa que contiene una declaración de procedimiento:

```
proc P( ▷x̄): if φ then S; P( ▷x̄) else skip fi do
                                                P( ▷x̄)
                                               od
```

donde $\bar{x}$ incluye a todas las variables que aparecen libres en S (cf. [Apt 81] o [Bak 80] para una definición de este concepto; téngase en cuenta, en todo caso, que en un programa de la forma proc $P(\bar{x}\triangleright\bar{y})$:S do S' od, cualquier aparición de un parámetro formal en S está ligada -no así en S'. De la misma forma, en un programa como dec x=t do S od, cualquier aparición de x en S está ligada -así como está ligada la aparición de x en x=t); obsérvese que dichas variables son pasadas como parámetros por referencia a fin de que S se ejecute sobre los valores de su ejecución previa.

La declaración anidada de, digamos, dos procedimientos mutuamente recursivos tal como proc $P...:S_1$, $Q...:S_2$ do S od puede expresarse mediante el siguiente programa:

```
proc P...: proc Q...:S2 do
                          S1
                         od
    do
      proc Q...:S2 do
                     S
                    od
    od
```

Evidentemente, cualquiera de las construcciones anteriores está sujeta a una semántica que, como veremos en la siguiente sección, hace que se comporten como razonablemente cabría esperar de ellas.

2.- HACIA UNA SEMANTICA DENOTACIONAL PARA UN LENGUAJE DE PROGRAMACION IMPERATIVO.

Como se dijo en la introducción de este capítulo nuestro objetivo es utilizar el formalismo del capítulo precedente para poder definir una semántica denotacional para los programas del lenguaje imperativo. La forma en que haremos ésto es caracterizar, de manera razonable, el comportamiento de los programas, y conseguir expresar dicho comportamiento mediante el formalismo funcional del que disponemos, definiendo apropiadamente la signatura y los tipos sobre los que se basen las expresiones que lo hagan.

La forma usual de definir la semántica de un programa es entender que éste representa una transformación (parcial) de estados, siendo un estado una aplicación que hace corresponder a cada variable, un dato; es decir, siendo un estado, una valoración de las variables (de individuo).

Si el lenguaje, como es el caso, dispone además de la posibilidad de realizar llamadas a procedimientos (declarados o no), entonces se añade a la semántica una valoración para las variables de procedimiento que asocie a cada una de ellas, una transformación adecuada. El tipo de transformación que se asocie depende de: (i) el carácter de los parámetros sobre los que se define (valor o referencia), (ii) el valor de los mismos y (iii) la forma en que comparten direcciones los parámetros por referencia (lo que suele llamarse "sharing" o relación de direccionamiento). Es evidente la dependencia de la semántica en los casos (i) y (ii). Respecto al tercero téngase en cuenta, por ejemplo, el siguiente programa:

proc P(▷x, y): if x=y then x:=1; x:=y fi do ... od

Una llamada P(▷x, y) con los contenidos de las direcciones de ambas, iguales, deja 1 en los contenidos de las direcciones de las dos, si dichas direcciones eran iguales, y no hace nada en otro caso. Por tanto no es igual, en principio, una llamada de la forma P(▷x, x) que una P(▷x, y), con y≠x.

Para expresar la manera en que una variable comparte una dirección con otra usaremos el concepto que denominamos *ubicación*. Una ubicación es una aplicación que asigna a cada variable de individuo, otra variable (posiblemente la misma); entonces diremos que la primera variable comparte la dirección de la segunda. Precisando estos conceptos consideremos la siguiente definición.

1.-Definición. *Estados, ubicaciones y contextos de procedimiento.*

Dada una signatura de primer orden Σ y una Σ-estructura $\mathfrak{D}$ definimos los conjuntos:

(1) el conjunto de los datos sobre $\mathfrak{D}$, $dat^{\mathfrak{D}}$, es el dominio D de la estructura $\mathfrak{D}$

(2) el conjunto de los estados (de memoria) sobre $\mathfrak{D}$, $est^{\mathfrak{D}}$, es el formado por las aplicaciones de la forma $\sigma: V \longrightarrow dat^{\mathfrak{D}}$

(3) el conjunto de las ubicaciones sobre $\mathfrak{D}$, $ubi^{\mathfrak{D}}$, es el formado por las aplicaciones de la forma $\tau: V \longrightarrow V$ tales que $\tau(x) = x$, casi para toda $x \in V$, esto es, para todas, salvo un número finito, de variables

(4) el conjunto de las denotaciones de procedimiento sobre $\mathfrak{D}$ para una variable (de procedimiento) P/mPn, $dpr^{\mathfrak{D}}(P)$, es el formado por las aplicaciones de la forma $\kappa: est^{\mathfrak{D}} \bullet (dat^{\mathfrak{D}})^m \bullet V^n \longrightarrow est^{\mathfrak{D}}$

(5) el conjunto de los contextos de procedimientos sobre $\mathfrak{D}$, $cpr^{\mathfrak{D}}$, es el formado por las aplicaciones ρ que asignan a cada variable de procedimiento una denotación sobre $\mathfrak{D}$ para ella. ■

Respecto a esta definición hay que hacer las siguientes observaciones. En primer lugar, pedimos que las ubicaciones solo reubiquen un número finito de variables. Esto coincide con lo que hace realmente un programa cualquiera S ya que éste solo contiene un número finito de variables y, en cada paso, solo puede reubicar a algunas de ellas; en todo caso, siempre un número finito.

En segundo lugar, las denotaciones de procedimiento toman como argumentos un estado, m datos y n variables. Entendemos que los m datos son los valores de los m parámetros actuales por valor y las n variables son los n parámetros actuales por referencia. Obsérvese que dichos argumentos permiten obtener la información de la que, según hemos dicho, depende una llamada a un procedimiento: los valores de los parámetros actuales por valor (a través de los m datos), la forma en que comparten dirección los parámetros actuales por referencia (a través de las n variables) y los contenidos de las direcciones de éstos últimos (a través del estado).

Sobre los conceptos que acabamos de definir podemos estudiar de qué manera un programa cualquiera S obtiene un estado, dados un contexto de procedimientos, una ubicación y otro estado. Hecho esto, la semántica de un programa se obtendrá tomando como ubicación inicial aquella que no ha reubicado ninguna variable todavía, esto es, la identidad.

Veamos entonces cómo definir la transformación parcial de estados que asociamos a cada programa S. Para ello estudiaremos algunos casos típicos empezando por la asignación.

Es bien sabido que en una asignación como x:=x+1, cada aparición de la variable x es entendida en un sentido diferente. La de la izquierda se refiere a la dirección de memoria donde está ubicada x y la de la derecha se refiere al contenido de dicha dirección. Por tanto, en presencia de un estado σ, una ubicación τ y un contexto de procedimiento ρ (irrelevante en este caso), el estado que resulta de dicha asignación es el que pone en la dirección de x, $\tau(x)$, el contenido de dicha dirección aumentado en una unidad, esto es $\sigma(\tau(x))+1$, y deja el resto de las direcciones de memoria con los contenidos que les asignaba σ.

Para expresar esto más formalmente introduzcamos alguna notación. Dado el conjunto A^B de aplicaciones Λ entre objetos de tipo A sobre objetos de tipo B, dados $a\in A$ y $b\in B$ definimos la aplicación $\Lambda[b/a]$ como aquella obtenida por:

$$\Lambda[b/a](x) = \begin{cases} b & \text{si } x=a \\ \Lambda(x) & \text{en otro caso} \end{cases}$$

para todo $x\in A$.

Según lo que acabamos de decir, el estado que obtenemos tras la asignación x:=t en presencia del estado σ y la ubicación τ es:

$$\sigma[t^{\mathcal{D}}(\sigma, \tau)/\tau(x)]$$

donde $t^{\mathcal{D}}(\sigma, \tau)$ es el dato que semánticamente hacemos corresponder al término t. Como hemos convenido en asignar a los contenidos de memoria, valores definidos, el estado anterior existirá si existe el dato correspondiente a t. Para calcular dicho dato podemos definir inductivamente sobre la estructura de t, la semántica de los términos en una estructura $\mathcal{D}$ bajo un estado σ y una ubicación τ. Tenemos pues:

$$c^{\mathcal{D}}(\sigma, \tau) = c^{\mathcal{D}}$$

$$x^{\mathcal{D}}(\sigma, \tau) = \sigma(\tau(x))$$

$$(f(t_1, \ldots, t_n))^{\mathcal{D}}(\sigma, \tau) = f^{\mathcal{D}}(t_1^{\mathcal{D}}(\sigma, \tau), \ldots, t_n^{\mathcal{D}}(\sigma, \tau)).$$

Pasemos ahora a ver cómo actúa la transformación de estados en el caso de la definición local de una variable. Sea pues un programa de la forma dec x=t do S od. Para la semántica de tales contrucciones seguimos lo que se conoce como *disciplina de ámbito estático* ("static scope"). Según ella, el valor que tenía la dirección de memoria de x antes de entrar en la declaración local dec, es recuperado tras salir de ella. Podríamos escribir entonces la semántica, en principio, como:

$$(S^{\mathcal{D}}(\sigma[t^{\mathcal{D}}(\sigma,\ \tau)/\tau(x)],\ \tau,\ \rho))[\sigma(\tau(x))/\tau(x)] \qquad (1)$$

Ahora bien supongamos que tenemos la siguiente declaración de procedimiento:

P(▷x): dec u=x+x do x:=u+u od

Para cualquier variable y, se espera intuitivamente que P(▷y) equivalga a y:=y+y+y+y. Sin embargo, si se aceptase (1), P(▷u) no tendría efecto. El problema es que el carácter local de u exige que u se "elija nueva", ésto es, se ubique en una dirección de memoria no usada por otra variable.

Dada entonces una ubicación τ, decimos que una variable x está *disponible* en τ si $\tau(x) = x$ y no existe ninguna otra variable y tal que $\tau(y) = x$. Obsérvese que, según se definieron las ubicaciones, existe siempre una cantidad infinita de variables disponibles.

En estas condiciones, la transformación de estados de la declaración local de una variable puede expresarse como sigue:

$$(S^{\mathcal{D}}(\sigma[t^{\mathcal{D}}(\sigma,\ \tau)/v],\ \tau[v/x],\ \rho))[\sigma(v)/v]$$

donde v es la primera variable nueva que no aparezca en dec x=t do S od y que esté disponible en τ.

Nótese que la disciplina de ámbito estático se mantiene para la variable v. Por otra parte nótese también que los cambios de contenidos de dirección de memoria que corresponden a v y que, en principio, deberían hacerse en $\tau(v)$, se hacen en v pues $\tau(v) = v$, por estar v disponible.

La introducción de una variable disponible evita hacer cualquier comentario sobre la sintaxis de las declaraciones de variables locales o de procedimientos, con vistas a imponer ciertas restricciones en ellas que traten de evitar los problemas que planteábamos más arriba.

Pasemos ahora a estudiar la transformación de estados para el caso de un programa con una declaración de procedimiento. Consideremos entonces proc $P(\bar{x}\triangleright\bar{y})$: S do S' od. Ya que nuestra intención es definir la semántica del lenguaje imperativo a través del formalismo funcional del capítulo anterior, entenderemos que el cuerpo S tendrá como semántica asociada el menor punto fijo de un cierto operador. Es por ello lógico decir entonces que:

$$(\text{proc } P(\bar{x}\triangleright\bar{y})\text{: S do S' od})^{\mathcal{D}} = (S')^{\mathcal{D}}(\sigma,\ \tau,\ \rho[\pi_0/P])$$

donde π_0 es una denotación de procedimiento sobre $\mathcal{D}$ para P obtenida a partir del cuerpo S como punto fijo de un operador apropiado T que transforma una denotación para P en otra.

En principio, el operador $T: dpr^{\mathcal{D}}(P) \longrightarrow dpr^{\mathcal{D}}(P)$ podría venir definido de la forma siguiente:

$$T(\pi)(\sigma', \bar{a}, \bar{z}) = (S[\bar{z}/\bar{y}])^{D}(\sigma'[\bar{a}/\bar{x}], \tau, \rho[\pi/P])$$

para cualesquiera estado σ', m-tupla de datos $\bar{a}$ y n-tupla de variables $\bar{z}$.

Sin embargo hacerlo así supondría definir una forma de sustitución que tuviera en cuenta los conceptos de compartición de memoria por variables; además, habría que hacer ciertos cambios con el fin de que se preserven los valores iniciales de $\bar{x}$. Todo ésto, como hemos visto en la declaración local, presenta algunos problemas.

En su lugar, para el tratamiento de los parámetros por referencia, manejaremos el concepto de ubicación de tal manera que la sustitución anterior se correspondería con una reubicación de las variables de $\bar{z}$ en las direcciones de las variables de $\bar{y}$. Para los parámetros por valor volvemos a considerar variables disponibles y realizamos sobre ellas todas las operaciones que correspondería hacer con los parámetros formales por valor.

Definimos pues el operador de la siguiente manera:

$$T(\pi)(\sigma', \bar{a}, \bar{z}) = (S^{D}(\sigma'[\bar{a}/\bar{u}], \tau', \rho[\pi/P]))[\sigma(\bar{u})/\bar{u}]$$

donde $\bar{u}$ es una n-tupla formada por las n primeras variables nuevas que no aparezcan en proc P($\bar{x}\triangleright\bar{y}$): S do S' od y estén disponibles en τ, y donde τ' es la ubicación que se obtiene a partir de la τ de la siguiente manera:

(i) $\tau'(y_j) = \tau(z_j)$, para todo $1 \leq j \leq n$

(ii) $\tau'(x_i) = u_i$, para todo $1 \leq i \leq m$

(iii) $\tau'(w) = \tau(w)$, en el resto de los casos.

Obsérvese que (i) se corresponde con la reubicación de los parámetros por referencia en las direciones de memoria de $\bar{z}$ mientras que (ii) se corresponde con la de los parámetros por valor en las direcciones de las variables nuevas (disponibles).

Como hicimos en la declaración local de variables, las variables disponibles recuperan su valor una vez ejecutado el cuerpo S de la declaración.

Por último, veamos cuál es el estado que resulta de la llamada a un procedimiento P. Dicho estado puede obtenerse de la manera siguiente:

$$(P(\bar{t}\triangleright\bar{y})^{D})(\sigma, \tau, \rho) = \rho(P)(\sigma, t_1^{D}(\sigma, \tau), \ldots, t_m^{D}(\sigma, \tau), \tau(y_1), \ldots, \tau(y_n))$$

Una vez que hemos estudiado la transformación de estados para los programas del lenguaje de comportamiento no usual (para el resto de los casos -skip, div, composición e instrucción condicional- es inmediato), como dijimos, la semántica de un programa cualquiera S puede ser definida como:

$$S^{D}(\sigma, \rho) = S^{D}(\sigma, id_V, \rho)$$

donde id_V es la identidad sobre el conjunto de variables.

3.- SEMANTICA DENOTACIONAL PARA UN LENGUAJE DE PROGRAMACION IMPERATIVO. OBJETIVOS.

Como dijimos a lo largo de la sección anterior, nuestro propósito es utilizar el formalismo que presentamos en el capítulo tercero, para definir la semántica del lenguaje imperativo, desde un punto de vista denotacional. Para ello basta tomar tipos elementales y una signatura sobre ellos apropiada de manera que sea posible expresar todas las transformaciones que estudiamos más arriba. Con tal fin haremos un estudio de los mismos programas que vimos entonces, tratando de encontrar los tipos y símbolos que serán necesarios.

Empezamos como siempre con la asignación. Recordemos que la semántica que pretendemos expresar es la transformación de estados definida por:

$$(x:=t)^{D}(\sigma,\ \tau,\ \rho) = \begin{cases} \sigma[t^{D}(\sigma,\ \tau)/\tau(x)] & \text{si } t^{D}(\sigma,\ \tau) \neq \perp_d \\ \perp_e & \text{en otro caso} \end{cases}$$

donde $\perp_d$ (resp. $\perp_e$) es el "bottom" del cpo plano que tiene por soporte a dat^{D} (resp. est^{D}).

Si pensamos en σ como una tupla infinita donde la componente i-ésima es el contenido de la dirección i-ésima, parece necesario disponer de un símbolo de función **asignar** que asigne datos a componentes de un estado de manera que podamos escribir asignar(σ, i, d), entendiendo que lo que queremos expresar es el estado resultante de asignar en la componente i-ésima de σ, el dato d.

A su vez, parece lógico disponer de otro símbolo de función **localizar** que averigüe la dirección en la que está ubicada una variable de tal forma que localizar(τ, x) represente la operación de ubicación $\tau(x)$.

Supongamos pues que disponemos de dichos símbolos. Una asignación como x:=x+1 podría expresarse como una función que transforma cada estado, ubicación y contexto de procedimientos, en otro estado. Consideremos pues los géneros *dat*, *est* y *ubi* y, para ser fieles a la notación que vinimos empleando en la sección anterior, representemos por σ y τ, las variables para los dos últimos tipos, respectivamente. Si la signatura contuviera el símbolo de función + de tipo funcional (*dat dat* $\longrightarrow$ *dat*) y la constante 1 de tipo *dat* entonces podríamos escribir la siguiente función definida por abstraccción:

$$\lambda\sigma\tau.\ \text{asignar}(\sigma,\ \text{localizar}(\tau,\ x),\ \sigma(\tau(x))+1)$$

Obsérvese que el contexto de procedimientos desaparece a nivel sintáctico. Como veremos más adelante, las variables de procedimiento tendrán su contrapartida, en el formalismo, en variables funcionales, y serán

tratadas, semánticamente, mediante valoraciones.

Quedaría únicamente por decir quién representa el papel de la variable x y cuáles son los tipos de los nuevos símbolos de función, a fin de que algo como $\sigma(\tau(x))$ tenga sentido. Para ello introducimos un nuevo género *var*. Con él podemos decir que asignar se interpreta como una función que obtiene un estado a partir de otro, una dirección y un dato, es decir, es un símbolo cuyo tipo funcional asociado es (*est var dat* $\longrightarrow$ *est*). Obsérvese que, como hicimos con las ubicaciones (cf. 2.1), identificamos direcciones de memoria con variables.

De manera análoga, localizar tendría como tipo asociado (*ubi var* $\longrightarrow$ *var*). Por último, para referirnos dentro del lenguaje a la variable x, supondremos que la signatura dispone de una cantidad numerable de constantes x de tipo *var* que se interpretarán como la propia variable x a la que están asociadas de manera natural.

Queda por decir cómo se expresa $\sigma(\tau(x))$. Sabemos que $\tau(x)$ se corresponde con localizar(τ, x). Parece necesario disponer de un símbolo de función valor de tipo asociado (*est var* $\longrightarrow$ *dat*), que permita obtener el dato que contiene una dirección de memoria en un estado de memoria. De esta forma podremos escribir entonces:

$$\text{valor}(\sigma, \text{localizar}(\tau, x))$$

para $\sigma(\tau(x))$, como buscábamos.

En consecuencia, hecho esto la expresión para la asignación x:=t podría ser la siguiente:

$$\lambda\sigma\tau.\ \text{asignar}(\sigma, \text{localizar}(\tau, x), t'(\sigma, \tau))$$

donde t' será una expresión funcional de tipo (*est ubi* $\longrightarrow$ *dat*) que puede ser definida por inducción sobre la estructura de t de la forma siguiente:

(i) $c' = \lambda\sigma\tau.\ c$

(ii) $x' = \lambda\sigma\tau.\ \text{valor}(\sigma, \text{localizar}(\tau, x))$

(iii) $(f(t_1, .., t_n)') = \lambda\sigma\tau.\ f(t_1'(\sigma, \tau), .., t_n'(\sigma, \tau))$

Nos ocupamos ahora de la declaración de variables locales. Recordemos, como antes, cuál era la semántica que queríamos asociarle y que venía dada por:

$$(\text{dec } x{=}t \text{ do } S \text{ od})^{\mathfrak{D}}(\sigma, \tau, \rho) = \begin{cases} S^{\mathfrak{D}}(\sigma[t^{\mathfrak{D}}(\sigma, \tau)/v], \tau[v/x], \rho)[\sigma(v)/v] \\ \qquad\qquad \text{si } t^{\mathfrak{D}}(\sigma, \tau) \neq \perp_d \\ \perp_e \qquad \text{en otro caso} \end{cases}$$

donde v es la primera variable que no aparece en la declaración y está disponible en τ.

Si suponemos que, para S, existe una expresión funcional S' de tipo asociado (*est ubi* → *est*), que expresa su semántica entonces, de las operaciones que aparecen más arriba, son expresables aquellas que se refieren a la modificación del estado de la forma en que se hacía para la asignación. Para $\tau[v/x]$, en cambio, parece necesario disponer de un símbolo de función ubicar de tipo asociado (*ubi var var* → *ubi*) que permita reubicar una variable en la dirección de otra dada. Con dicho símbolo podemos escribir ubicar(τ, x, v) para expresar $\tau[v/x]$.

Sin embargo, lo que no podemos expresar es el hecho de que v sea la primera variable disponible. Para hacerlo supondremos que disponemos de un índice que nos diga, en cada momento, cuál es la primera variable que no ha sido reubicada ni aparece en el programa y está, en consecuencia, disponible. Por tanto introducimos un nuevo tipo *nat* para poder referirnos a dicho índice, y añadimos el símbolo de función s (de sucesor) y la constante 0, para poder expresarlo.

En estas condiciones, la expresión funcional que asociamos para un programa S obtendrá un estado a partir de otro dado, de una ubicación y de un índice. Es decir, el tipo funcional asociado a dicha expresión habrá de ser (*est ubi nat* → *est*). Obsérvese que si ι es el índice en un momento dado, parece necesario disponer de un símbolo de función var de tipo (*nat* → *var*) de forma que var(ι) sea la primera variable disponible.

La expresión correspondiente a lo que en la semántica de la declaración local de variables se escribe mediante:

$$S^{\mathcal{D}}(\sigma[t^{\mathcal{D}}(\sigma, \tau)/v], \tau[v/x], \rho)$$

podemos ahora escribirla con $F(\sigma, \tau, \iota)$ de tipo *est* definida por:

$$S'(\text{asignar}(\sigma, \text{var}(\iota), t'(\sigma, \tau)), \text{ubicar}(\tau, x, \text{var}(\iota)), s(\iota))$$

siendo S' la expresión inductivamente definida, correspondiente a S.

Obsérvese que, como se ha tomado la variable disponible var(ι), el índice debe aumentar en una unidad, lo que hacemos con s(ι).

Con $F(\sigma, \tau, \iota)$ podemos escribir la siguiente expresión para la declaración local de variables, sin más que devolver al contenido de la dirección de la variable disponible, su valor original. Es decir:

$$\lambda\sigma\tau\iota.\ \text{asignar}(F(\sigma, \tau, \iota), \text{var}(\iota), \text{valor}(\sigma, \text{var}(\iota)))$$

La introducción de un nuevo argumento en la expresión funcional no afecta a la que dimos para la asignación. Se puede escribir simplemente:

$$\lambda\sigma\tau\iota.\ \text{asignar}(\sigma, \text{localizar}(\tau, x), t'(\sigma, \tau))$$

Pasamos ahora a la declaración de procedimientos. Como en los casos anteriores, recordemos que la semántica de proc $P(\bar{x}\triangleright\bar{y})$: S_0 do S_1 od se entiende como:

$$(\text{proc } P(\bar{x}\triangleright\bar{y})\text{: } S_0 \text{ do } S_1 \text{ od})^{\mathcal{D}}(\sigma, \tau, \rho) = S_1^{\mathcal{D}}(\sigma, \tau, \rho[\pi_0/P])$$

donde π_0 es la denotación de procedimiento sobre $\mathcal{D}$ para P obtenido como punto fijo del operador $T: dpr^{\mathcal{D}} \longrightarrow dpr^{\mathcal{D}}$ definido por:

$$T(\pi)(\sigma', \bar{a}, \bar{z}) = (S_0^{\mathcal{D}}(\sigma'[\bar{a}/\bar{u}], \tau[\tau(\bar{z})/\bar{y}, \bar{u}/\bar{x}], \rho[\pi/P]))[\sigma(\bar{u})/\bar{u}]$$

donde $\bar{u}$ es una n-tupla formada por las n primeras variables que no aparecen en la declaración y están disponibles en τ.

Para expresar el punto fijo utilizamos el operador de recursión μ de nuestro lenguaje. Dada la variable de procedimiento P/m▷n, le asociamos la variable funcional X_τ de tipo τ = (*est dat* $\overset{m}{\ldots}$ *dat var* $\overset{n}{\ldots}$ *var* $\longrightarrow$ *est*) (no confundir el tipo τ con las variables de tipo *ubi*). Entonces el punto fijo es el de la función que expresa el comportamiento del operador anterior. Dicha función, que toma como argumentos un estado σ, m variables $d_1, \ldots, d_m$ de tipo *dat* y n variables $v_1, \ldots, v_n$ de tipo *var*, se obtiene a través de los siguientes pasos:

(i) para $\sigma[\bar{d}/\bar{u}]$ utilizamos la expresión obtenida como sigue. Definamos la sucesión de estados:

$$\sigma_i = \text{asignar}(\sigma_{i-1}, d_{m-i+1}, \text{var}(s^{m-i+1}(\iota)))$$

para i=1, .., m, siendo $\sigma_0 = \sigma$. Entonces la expresión que buscamos es σ_m. Obsérvese que la asignación secuencial no es relevante por ser las variables de $\bar{u}$, disponibles y distintas

(ii) para $\tau[\tau(\bar{v})/\bar{y}, \bar{u}/\bar{x}]$ utilizamos la expresión obtenida a través de la siguiente sucesión de ubicaciones:

(1) para $\bar{u}/\bar{x}$ tenemos:

$$\tau'_i = \text{ubicar}(\tau'_{i-1}, x_{m-i+1}, \text{var}(s^{m-i+1}(\iota)))$$

para i=1, .., m, siendo $\tau'_0 = \tau$

(2) obtenida la ubicación τ'_m, para $\tau(\bar{v})/\bar{y}$ hacemos:

$$\tau_j = \text{ubicar}(\tau_{j-1}, y_{n-j+1}, \text{localizar}(\tau_{j-1}, v_{n-j+1}))$$

para j=1, .., n, siendo $\tau_0 = \tau'_m$. La expresión que buscamos es entonces τ_n.

Como en (i), obsérvese que la ubicación secuencial no es relevante pues las variables de $\bar{x}$ e $\bar{y}$, son todas distintas, según definimos la sintaxis, y las de $\bar{u}$ son distintas de las anteriores, disponibles y distintas entre sí

(iii) por último, para $[\sigma(\bar{u})/\bar{u}]$, tenemos la siguiente sucesión de estados:

$$\sigma'_i = \text{asignar}(\sigma'_{i-1}, \text{var}(s^{m-i+1}(\iota)), \text{valor}(\sigma, \text{var}(s^{m-i+1}(\iota))))$$

para $i=1, \ldots, m$, siendo $\sigma'_0 = S'_0(\sigma_m, \tau_n, s^{m+1}(\iota))$ y S'_0, la expresión correspondiente a S_0. Valen de nuevo los mismos comentarios sobre la irrelevancia de la asignación secuencial.

Una vez obtenida la expresión del apartado (iii), podemos expresar el comportamiento del operador dado por:

$$T(\pi)(\sigma', \bar{a}, \bar{z}) = (S_0^{\mathfrak{D}}(\sigma'[\bar{a}/\bar{u}], \tau[\tau(\bar{z})/\bar{y}, \bar{u}/\bar{x}], \rho[\pi/P]))[\sigma(\bar{u})/\bar{u}]$$

mediante la expresión funcional definida a través del operador de recursión μ como:

$$\mu X_\tau.\lambda\sigma\ d_1 \ldots d_m\ v_1 \ldots v_n\ \iota.\ \sigma'_m$$

En consecuencia, la expresión con la que podemos definir la semántica de la declaración de procedimientos puede ser obtenida sustituyendo cada aparición de P (esto es, de la variable funcional X_τ), por la expresión que denota su punto fijo. Por tanto, la semántica que venía dada por:

$$(\text{proc } P(\bar{x}\triangleright\bar{y})\text{: } S_0 \text{ do } S_1 \text{ od})^{\mathfrak{D}}(\sigma, \tau, \rho) = S_1^{\mathfrak{D}}(\sigma, \tau, \rho[\pi_0/P])$$

puede ser expresada con:

$$\lambda\sigma\tau\iota.\ S'_1[\mu X_\tau.\lambda\sigma\ d_1 \ldots d_m\ v_1 \ldots v_n\ \iota.\ \sigma'_m/X_\tau](\sigma, \tau, \iota)$$

siendo S'_1 la expresión correspondiente a S_1.

Obsérvese que, por el lema de sustitución §3.4.5, la sustitución que realizamos equivale a considerar, semánticamente, la variable de procedimiento P como su punto fijo.

Finalmente nos ocupamos de las llamadas a procedimientos. Recordemos que la semántica de tales construcciones debe ser:

$$(P(\bar{t}\triangleright\bar{y}))^{\mathfrak{D}})(\sigma, \tau, \rho) = \rho(P)(\sigma, t_1^{\mathfrak{D}}(\sigma, \tau), \ldots, t_m^{\mathfrak{D}}(\sigma, \tau), \tau(y_1), \ldots, \tau(y_n))$$

siendo la aridad de P, $m \triangleright n$. Consideremos pues la variable funcional X_τ de tipo $\tau = (est\ dat \overset{m}{\ldots} dat\ var \overset{n}{\ldots} var \rightarrow est)$, entonces la expresión que buscamos es:

$$X_\tau(\sigma, t'_1(\sigma, \tau), \ldots, t'_m(\sigma, \tau), \text{localizar}(\tau, y_1), \ldots, \text{localizar}(\tau, y_n))$$

4.- FORMALIZACION DE LA SEMANTICA DENOTACIONAL.

Una vez encontradas las expresiones que denotan la semántica que esperamos de los programas, resumimos todos los conceptos formalmente. Empezamos pues considerando los tipos elementales y las signaturas.

1.-Definición. *Tipos elementales y signaturas.*

El conjunto de los tipos elementales está formado por los géneros *dat*, *var*, *nat*, *est* y *ubi*.

Dada una signatura de primer orden Σ para el conjunto de programas recursivos P_Σ, definimos la signatura heterogénea Σ' sobre el conjunto de géneros anterior, formada por los siguientes símbolos:

(1) Constantes:

(i) c de género *dat*, para cada constante $c \in \Sigma$

(ii) x_n de género *var*, para cada $n \in N$

(iii) 0 de género *nat*

(2) Símbolos de función:

(i) f de tipo (funcional) $(dat \overset{n}{\ldots} dat \longrightarrow dat)$, para cada símbolo de función n-ario $f \in \Sigma$

(ii) s de tipo $(nat \longrightarrow nat)$

(iii) asignar de tipo $(est\ var\ dat \longrightarrow est)$

(iv) ubicar de tipo $(ubi\ var\ var \longrightarrow ubi)$

(v) valor de tipo $(est\ var \longrightarrow dat)$

(vi) localizar de tipo $(ubi\ var \longrightarrow var)$

(vii) var de tipo $(nat \longrightarrow var)$

(3) Símbolos de predicado: R de tipo (relacional) $(dat \overset{n}{\ldots} dat)$, para cada símbolo de relación n-ario $R \in \Sigma$. ∎

La interpretación de los símbolos anteriores está determinada por las expresiones que obtuvimos más arriba.

2.-Definición. *Σ'-estructuras standard.*

Una Σ'-estructura standard $\mathfrak{D}$ está formada por los siguientes dominios e interpretaciones de símbolos de Σ':

(1) Dominios:

D^{dat} es el cpo sobre un soporte de género *dat*

$D^{var} = V_\perp$ (cpo plano con el conjunto de las variables (de individuo) del lenguaje como soporte de *var*)

$D^{nat} = N_\perp$ (cpo plano con el conjunto de los números naturales como soporte de *nat*)

$D^{ubi} = ubi_\perp^{\mathfrak{D}}$ (cpo plano con el conjunto de ubicaciones sobre $\mathfrak{D}$ (cf. 2.1.(3)) como soporte de *ubi*)

$D^{est} = est_{\perp}^{\mathfrak{D}}$ (cpo plano con el conjunto de estados sobre $\mathfrak{D}$ (cf. 2.1.(2)) como soporte de *est*).

(2) Interpretaciones de símbolos:

(i) Constantes:

$c^{\mathfrak{D}} \in D^{dat}$

$x^{\mathfrak{D}} = x$ (la correspondiente variable de V)

$0^{\mathfrak{D}} = 0$ (cero de los números naturales).

(ii) Símbolos de función:

$f^{\mathfrak{D}}: D^{dat} \bullet \overset{n}{\ldots} \bullet D^{dat} \to D^{dat}$ es una función estricta

$s^{\mathfrak{D}}: D^{nat} \to D^{nat}$ es la función estricta definida como el sucesor de números naturales, esto es, $s^{\mathfrak{D}}(n) = n+1$, para todo número natural $n \in \mathbb{N}$

$\mathbf{asignar}^{\mathfrak{D}}: D^{est} \bullet D^{var} \bullet D^{dat} \to D^{est}$ es una función estricta definida por:

$$\mathbf{asignar}^{\mathfrak{D}}(\sigma, x, d) = \sigma[d/x]$$

para todo estado $\sigma \in est^{\mathfrak{D}}$, variable $x \in V$ y dato $d \in dat^{\mathfrak{D}}$

$\mathbf{ubicar}^{\mathfrak{D}}: D^{ubi} \bullet D^{var} \bullet D^{var} \to D^{ubi}$ es una función estricta definida por:

$$\mathbf{ubicar}^{\mathfrak{D}}(\tau, x, y) = \tau[y/x]$$

para toda ubicación $\tau \in ubi^{\mathfrak{D}}$ y variables $x, y \in V$

$\mathbf{valor}^{\mathfrak{D}}: D^{est} \bullet D^{var} \to D^{dat}$ es una función estricta definida por:

$$\mathbf{valor}^{\mathfrak{D}}(\sigma, x) = \sigma(x)$$

para todo estado $\sigma \in est^{\mathfrak{D}}$ y variable $x \in V$

$\mathbf{localizar}^{\mathfrak{D}}: D^{ubi} \bullet D^{var} \to D^{var}$ es una función estricta definida por:

$$\mathbf{localizar}^{\mathfrak{D}}(\tau, x) = \tau(x)$$

para toda ubicación $\tau \in ubi^{\mathfrak{D}}$ y variable $x \in V$

$\mathbf{var}^{\mathfrak{D}}: D^{nat} \to D^{var}$ es una función estricta definida por:

$$\mathbf{var}^{\mathfrak{D}}(n) = x_n$$

para todo número natural $n \in \mathbb{N}$. Recuérdese que suponemos que cada $x \in V$ es de la forma x_n, siendo $n \in \mathbb{N}$ un índice para x, unívocamente determinado.

(iii) Símbolos de predicado:

$R^{\mathfrak{D}}: D^{dat} \bullet \overset{n}{\ldots} \bullet D^{dat} \to B$ es una función estricta. ∎

Como venimos explicando a lo largo de esta sección, nuestro objetivo es utilizar el formalismo que introdujimos en el capítulo anterior para definir la semántica de los programas recursivos que introdujimos en éste.

La semántica de un programa S cualquiera se define a través de dos etapas. En la primera, definimos de qué manera obtiene S un nuevo estado a partir de otro dado, una ubicación y un índice que señala a la primera variable disponible. Esto es, definimos cuál es el significado de la *ejecución* del programa S. Para ello utilizamos las expresiones que obtuvimos más arriba.

En la segunda, definimos la semántica, propiamente dicha, de S, considerando, como ubicación inicial, la identidad sobre el conjunto de variables (es decir, considerando una ubicación que no ha reubicado ninguna variable todavía) y $s^k(0)$, como índice de la primera variable disponible, siendo k el menor número natural tal que toda variable que aparezca en S tiene un índice menor que k. En definitiva, definimos el *significado* del programa S.

3.-<u>Definición</u>. *Ejecución de programas.*

La ejecución de un programa S es una expresión funcional $EJEC_S \in EF^{\tau}_{\Sigma}$, con τ = (*est ubi nat* $\longrightarrow$ *est*), definida por inducción sobre la estructura de S como sigue:

(i) S = skip: $EJEC_S = \lambda\sigma\tau\iota.\ \sigma$

(ii) S = div: $EJEC_S = \lambda\sigma\tau\iota.\ \Omega_\tau(\sigma)$, donde τ es el tipo funcional (*est* $\longrightarrow$ *est*)

(iii) S = x:=t:

$EJEC_S = \lambda\sigma\tau\iota.$ asignar(σ, localizar(τ, x), $E_t(\sigma, \tau)$)

donde E_t se define por inducción sobre la estructura de t de la siguiente manera:

(a) t = c: $E_t = \lambda\sigma\tau.$ c

(b) t = x: $E_t = \lambda\sigma\tau.$ valor(σ, localizar(τ, x))

(c) $t = f(t_1, \ldots, t_n)$: $E_t = \lambda\sigma\tau.\ f(E_{t1}(\sigma, \tau), \ldots, E_{tn}(\sigma, \tau))$

(iv) $S = S_1; S_2$: $EJEC_S = \lambda\sigma\tau\iota.\ EJEC_{S2}(EJEC_{S1}(\sigma, \tau, \iota), \tau, \iota)$

(v) S = if φ then S_1 else S_2 fi:

$EJEC_S = \lambda\sigma\tau\iota.\ E_\varphi(\sigma, \tau) \longrightarrow EJEC_{S1}(\sigma, \tau, \iota), EJEC_{S2}(\sigma, \tau, \iota)$

donde $E_\varphi(\sigma, \tau)$ es una fórmula que se define por inducción sobre la estructura de φ, de la siguiente manera:

(a) $\varphi = R(t_1, \ldots, t_n)$: $E_\varphi(\sigma, \tau) = R(E_{t1}(\sigma, \tau), \ldots, E_{tn}(\sigma, \tau))$

(b) $\varphi = t_1 \simeq t_2$: $E_\varphi(\sigma, \tau) = E_{t1}(\sigma, \tau) \simeq E_{t2}(\sigma, \tau)$

(c) $\varphi = \neg\varphi_1$: $E_\varphi(\sigma, \tau) = \neg E_{\varphi 1}(\sigma, \tau)$

(d) $\varphi = \varphi_1 \wedge \varphi_2$: $E_\varphi(\sigma, \tau) = E_{\varphi 1}(\sigma, \tau) \wedge E_{\varphi 2}(\sigma, \tau)$

(vi) S = dec x=t do S_1 od:

$EJEC_S = \lambda\sigma\tau\iota.$ asignar(F(σ, τ, ι), var(ι), valor(σ, var(ι)))

donde la expresión F(σ, τ, ι) viene dada por:

$EJEC_{S1}$(asignar(σ, var(ι), $E_t(\sigma, \tau)$), ubicar(τ, x, var(ι)), s(ι))

(vii) S = proc P($\bar{x}\triangleright\bar{y}$): S_0 do S_1 od:

$EJEC_S = \lambda\sigma\tau\iota.\ EJEC_{S1}[\mu X_\tau.\lambda\sigma\ d_1 \ldots d_m\ v_1 \ldots v_n\ \iota.\ \sigma'_m/X_\tau](\sigma, \tau, \iota)$

donde σ'_m es una expresión de tipo *est* obtenida como último elemento de la siguiente sucesión de expresiones de dicho tipo:

$$\sigma'_i = \text{asignar}(\sigma'_{i-1}, \text{var}(s^{m-i+1}(\iota)), \text{valor}(\sigma, \text{var}(s^{m-i+i}(\iota))))$$

para todo i=1, ..., m, donde:

$$\sigma'_0 = EJEC_{S0}(\sigma_m, \tau_n, s^{m+1}(\iota))$$

siendo σ_m el último elemento de la sucesión dada por:

$$\sigma_i = \text{asignar}(\sigma_{i-1}, d_{m-i+1}, \text{var}(s^{m-i+1}(\iota)))$$

para todo i=1, ..., m, siendo $\sigma_0 = \sigma$, y siendo, a su vez, τ_n el último elemento de la sucesión:

$$\tau_j = \text{ubicar}(\tau_{j-1}, y_{n-j+1}, \text{localizar}(\tau_{j-1}, v_{n-j+1}))$$

para todo j=1, ..., n, donde $\tau_0 = \tau'_m$ siendo τ'_m, por su parte, el último elemento de la sucesión dada por:

$$\tau'_i = \text{ubicar}(\tau'_{i-1}, x_{m-i+1}, \text{var}(s^{m-i+1}(\iota)))$$

para i=1, ..., m, donde $\tau'_0 = \tau$.

(viii) S = P($\bar{t}\triangleright\bar{y}$):

$EJEC_S$ =

$= X_\tau(\sigma, E_{t1}(\sigma, \tau), \ldots, E_{tm}(\sigma, \tau), \text{localizar}(\tau, y_1), \ldots, \text{localizar}(\tau, y_n))$

■

Respecto al significado de un programa S, según se dijo más arriba, el índice de la primera variable disponible debería ser, en principio, $s^k(0)$, donde k es el menor número tal que toda variable que aparezca en S es de la forma x_n, con n<k; para la ubicación tomábamos aquella que hasta ese momento

no había reubicado ninguna variable, esto es, la identidad. La expresión $\lambda v.\ v$, siendo v una variable de tipo *var*, es la función identidad entre variables, sin embargo no es un objeto de tipo *ubi*, como necesitamos. Para poder referirnos a ello introducimos un nuevo, y último, símbolo a la signatura Σ', que ya teníamos. Dicho símbolo es una constante id_v de tipo *ubi*, cuya interpretación en una estructura $\mathcal{D}$ es la obvia, es decir, $id_v^{\mathcal{D}} \in ubi_\perp^{\mathcal{D}}$ está definida como una función estricta tal que $id_v^{\mathcal{D}}(v) = v$, para toda variable $v \in V$.

En estas condiciones se puede definir el significado de un programa de la siguiente manera.

4.-<u>Definición</u>. *Significado de un programa.*

El significado de un programa S es una expresión funcional $SIG_S \in EF_{\Sigma'}^{\tau}$, con $\tau = (est \longrightarrow est)$, definida por:

$$SIG_S = \lambda\sigma.\ EJEC_S(\sigma,\ id_v,\ s^k(0))$$

siendo k el menor número natural tal que toda variable que aparezca en S tiene un índice menor que k. ∎

5.- TRADUCCION DE UNA LOGICA DINAMICA DE PROGRAMAS RECURSIVOS.

Una vez que disponemos de una semántica (denotacional) para los programas recursivos expresada en el formalismo del capítulo anterior (esto es, LFRP) podemos traducir las fórmulas de la lógica dinámica F_Σ que veíamos en 1.1, a la lógica de primer orden para funciones recursivas parciales LFRP. Antes de ello hemos de definir cómo se interpretan ciertas fórmulas como $\langle S\rangle\varphi$ en un contexto que disponga de tres valores booleanos.

Es conocido que $\langle S\rangle\varphi$, en lógica dinámica ciásica (cf. [Har 79], para QDL), se hace cierta bajo un estado de una cierta estructura, si φ, a su vez, se hace cierta en el estado resultante (si el lenguaje dispone de una construcción no determinista como QDL hablaríamos de "algún estado resultante") de la ejecución de S, si es que existe, es decir, si es que S no diverge. Cuando S pueda tomar como valor el elemento indefinido $\perp_e$ del cpo plano de estados entonces es lógico pedir que la fórmula en cuestión, $\langle S\rangle\varphi$, se haga falsa. Así lo haremos, tal como indicamos en la siguiente definición.

1.-Definición. *Semántica de las fórmulas de* F_Σ.

Dada una Σ'-estructura standard $\mathfrak{D}$, un contexto de procedimientos ρ y un estado $\underline{\sigma}\in est^{\mathfrak{D}}$, definimos la semántica de las fórmulas $\varphi\in F_\Sigma$, por inducción sobre su estructura como sigue:

(1) $(\mathfrak{D}, \underline{\sigma}, \rho)(t_1 \propto t_2)$ tiene por semántica la que se ha visto en capítulos anteriores, siendo la semántica de los términos la dada por:

(i) $(\mathfrak{D}, \underline{\sigma}, \rho)(c) = c^{\mathfrak{D}}$

(ii) $(\mathfrak{D}, \underline{\sigma}, \rho)(x) = \underline{\sigma}(x)$

(iii) $(\mathfrak{D}, \underline{\sigma}, \rho)(f(t_1, \ldots, t_n)) = f^{\mathfrak{D}}((\mathfrak{D}, \underline{\sigma}, \rho)(t_1), \ldots)$

(2) La semántica en los casos de la negación y la disyunción se define como en LFRP

(3) $$(\mathfrak{D}, \underline{\sigma}, \rho)(\exists x\varphi) = \begin{cases} \underline{t} & \text{si existe } d\in D \text{ tal que } (\mathfrak{D}, \underline{\sigma}[d/x], \rho)(\varphi)=\underline{t} \\ \underline{f} & \text{si } (\mathfrak{D}, \underline{\sigma}[d/x], \rho)(\varphi)=\underline{f}, \text{ para todo } d\in D \\ \perp & \text{en otro caso} \end{cases}$$

(5) $$(\mathfrak{D}, \underline{\sigma}, \rho)(\langle S\rangle\varphi) = \begin{cases} (\mathfrak{D}, \underline{\sigma}', \underline{\rho})(\varphi) & \text{si } \perp_e \neq \underline{\sigma}' = (\mathfrak{D}, \underline{\sigma}, \rho)(S) \\ \underline{f} & \text{en otro caso} \end{cases}$$

donde $(\mathfrak{D}, \underline{\sigma}, \rho)(S)$ es la semántica dada por $(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma](SIG_S(\sigma))$. ■

Definida la semántica de las fórmulas de la lógica dinámica para programas recursivos veamos que existe una traducción de ellas a fórmulas de LFRP de manera que el valor booleano de ambas sea el mismo. Para ello empezamos demostrando que, tal como se ha definido, el significado de los términos $t\in T_\Sigma$, programas $S\in P_\Sigma$ y fórmulas de primer orden libres de cuantificadores $\varphi\in F_\Sigma$ de 1.1 tienen el mismo comportamiento semántico que E_t, SIG_S y E_φ, respectivamente.

2.-Teorema.

Dada una Σ'-estructura standard $\mathfrak{D}$, un estado $\underline{\sigma}\in est^{\mathfrak{D}}$ y un contexto de procedimientos ρ se verifica que:

(a) $(\mathfrak{D}, \underline{\sigma}, \rho)(t) = (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](E_t(\sigma, \tau))$

(b) $(\mathfrak{D}, \underline{\sigma}, \rho)(S) = (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma](SIG_S(\sigma))$

(c) $(\mathfrak{D}, \underline{\sigma}, \rho)(\varphi) = (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](E_\varphi(\sigma, \tau))$.

Demostración. (a) se demuestra por inducción sobre la estructura de t. Por ejemplo, para el caso en que t es una variable tenemos:

$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](E_x) =$

$= (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](valor(\sigma, localizar(\tau, x))) =$

$= valor^{\mathfrak{D}}(\underline{\sigma}, localizar^{\mathfrak{D}}(id_V^{\mathfrak{D}}, x^{\mathfrak{D}})) =$

$= \underline{\sigma}(x) = (\mathfrak{D}, \underline{\sigma}, \rho)(x)$

(b) Es evidente tal y como se ha definido la semántica de los programas.

(c) Es inmediato para el caso en que φ es una igualdad o un símbolo de predicado aplicado a n términos, teniendo en cuenta el apartado (a), para éstos. El resto de los casos es inmediato, por hipótesis de inducción. ■

Una vez visto en qué sentido hablamos de traducción de fórmulas y preservación de la semántica, pasamos a demostrar que ésta se mantiene también, no solo para las de primer orden libres de cuantificadores.

3.-Teorema. *Traducción de la lógica dinámica a LFRP.*

Dada una fórmula $\varphi \in F_\Sigma$ existe una Σ'-fórmula $\varphi'(\sigma, \tau)$ con una variable σ de tipo *est* y otra τ de tipo *ubi*, libres, de manera que, para toda Σ'-estructura standard $\mathfrak{D}$, estado $\underline{\sigma} \in est^{\mathfrak{D}}$ y contexto de procedimientos ρ se verifica que:

$$(\mathfrak{D}, \underline{\sigma}, \rho)(\varphi) = (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](\varphi'(\sigma, \tau)).$$

Demostración. Definamos la fórmula φ' por inducción sobre la estructura de φ:

(i) $(t_1 \simeq t_2)' = E_{t1}(\sigma, \tau) \simeq E_{t2}(\sigma, \tau)$

(ii) $(R(t_1, \ldots, t_n))' = R(E_{t1}(\sigma, \tau), \ldots, E_{tn}(\sigma, \tau))$

(iii) $(\neg\varphi)' = \neg\varphi'(\sigma, \tau)$

(iv) $(\varphi_1 \vee \varphi_2)' = \varphi_1'(\sigma, \tau) \vee \varphi_2'(\sigma, \tau)$

(v) $(\exists x\varphi)' = \exists x^{dat}\varphi'(\sigma, \tau)$

(vi) $(\langle S \rangle\varphi)' = \exists \sigma'^{est}[\Delta\, SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma) \wedge \varphi'(\sigma', \tau)]$

Salvo para $\langle S \rangle\varphi$, demostraremos, sin pérdida de generalidad, solamente el caso en que el valor booleano de la fórmula en cuestión es $\underline{t}$. De forma similar se demostraría en el resto de los casos.

Tenemos entonces que:

(i) Si $(\mathfrak{D}, \underline{\sigma}, \rho)(t_1 \simeq t_2) = \underline{t}$ entonces resulta que:

$$\perp_d \neq (\mathfrak{D}, \underline{\sigma}, \rho)(t_1) = (\mathfrak{D}, \underline{\sigma}, \rho)(t_2) \neq \perp_d$$

Por el teorema anterior tenemos que:

$$\perp_d \neq (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](E_{t1}(\sigma, \tau)) = (\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](E_{t2}(\sigma, \tau)) \neq \perp_d$$

luego el valor semántico de la igualdad es el mismo en ambos casos. El caso (ii) para los símbolos de predicado es análogo.

(iii)-(iv) Los casos para la negación y la disyunción son inmediatos, por hipótesis de inducción.

(v) Si $(\mathfrak{D}, \underline{\sigma}, \rho)(\exists x\varphi) = \underline{t}$ entonces:

$$(\mathfrak{D}, \underline{\sigma}[d/x], \rho)(\varphi) = \underline{t}$$

para algún dato definido $d \in D$. Por tanto:

$$(\mathfrak{D}, \rho)[\underline{\sigma}[d/x]/\sigma, id_V^{\mathfrak{D}}/\tau](\varphi'(\sigma, \tau)) = \underline{t}$$

por hipótesis de inducción. Entonces, por el lema de coincidencia §3.4.3:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau, d/x^{dat}](\varphi'(\sigma, \tau)) = \underline{t}$$

luego:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](\exists x^{dat}\varphi'(\sigma, \tau)) = \underline{t}$$

como queríamos demostrar.

(vi) Si $(\mathfrak{D}, \underline{\sigma}, \rho)(\langle S\rangle\varphi) = \underline{t}$ entonces:

$$\perp_e \neq \underline{\sigma}' = (\mathfrak{D}, \underline{\sigma}, \rho)(S) \text{ y } (\mathfrak{D}, \underline{\sigma}', \rho)(\varphi) = \underline{t}$$

Por el teorema anterior tendremos entonces que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma](SIG_S(\sigma)) = \underline{\sigma}' \neq \perp_e$$

con lo que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, \underline{\sigma}'/\sigma'](\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma)) = \underline{t}$$

y, por el lema de coincidencia:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau, \underline{\sigma}'/\sigma'](\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma)) = \underline{t} \qquad (1)$$

Por otra parte, por hipótesis de inducción:

$$(\mathfrak{D}, \rho)[\underline{\sigma}'/\sigma', id_V^{\mathfrak{D}}/\tau](\varphi'(\sigma', \tau)) = \underline{t}$$

luego, por el lema de coincidencia:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau, \underline{\sigma}'/\sigma'](\varphi'(\sigma', \tau)) = \underline{t} \qquad (2)$$

De (1) y (2) resulta que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau, \underline{\sigma}'/\sigma'](\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma) \wedge \varphi'(\sigma', \tau)) = \underline{t}$$

con lo que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma, id_V^{\mathfrak{D}}/\tau](\exists\sigma'^{est}(\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma) \wedge \varphi'(\sigma', \tau))) = \underline{t}$$

como buscábamos.

Si $(\mathfrak{D}, \underline{\sigma}, \rho)(\langle S\rangle\varphi) = \underline{f}$ entonces caben dos posibilidades:

(a) $(\mathfrak{D}, \underline{\sigma}, \rho)(S) = \perp$. Por el teorema anterior resulta que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma](SIG_S(\sigma)) = \perp$$

luego $(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma](\Delta\ SIG_S(\sigma)) = \underline{f}$. En consecuencia, para todo estado $\underline{\sigma}'$ tenemos, por el lema de coincidencia, que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau,\ \underline{\sigma}'/\sigma'](\sigma' \simeq SIG_S(\sigma) \wedge \Delta\ SIG_S(\sigma) \wedge \varphi'(\sigma',\ \tau)) = \underline{f}$$

luego:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau](\exists\sigma' \in^{est} (\sigma' \simeq SIG_S(\sigma) \wedge \Delta\ SIG_S(\sigma) \wedge \varphi'(\sigma',\ \tau))) = \underline{f}$$

como queríamos demostrar.

(b) $(\mathfrak{D}, \underline{\sigma}, \rho)(S) = \underline{\sigma}'$ y $(\mathfrak{D}, \underline{\sigma}', \rho)(\varphi) = \underline{f}$. Por el teorema anterior tenemos que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma](SIG_S(\sigma)) = \underline{\sigma}' \neq \perp$$

con lo que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau,\ \underline{\sigma}''/\sigma'](\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma)) \neq \perp \qquad (1)$$

para todo estado $\underline{\sigma}'' \in est^{\mathfrak{D}}$. Obsérvese que para $\underline{\sigma}'$ la fórmula es cierta y para el resto de los estados, es falsa.

Como, por hipótesis de inducción, tenemos que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau,\ \underline{\sigma}'/\sigma'](\varphi'(\sigma',\ \tau)) = \underline{f} \qquad (2)$$

entonces, de (1) y (2), concluimos:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau,\ \underline{\sigma}''/\sigma'](\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma) \wedge \varphi'(\sigma',\ \tau)) = \underline{f}$$

para todo $\underline{\sigma}'' \in est^{\mathfrak{D}}$. Obsérvese que para $\underline{\sigma}'$ se hace falsa "$\varphi'(\sigma',\ \tau)$" y, para el resto, se hace falsa "$\Delta\ SIG_S(\sigma) \wedge \sigma' \simeq SIG_S(\sigma)$". Por tanto:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau](\exists\sigma' \in^{est} (\sigma' \simeq SIG_S(\sigma) \wedge \Delta\ SIG_S(\sigma) \wedge \varphi'(\sigma',\ \tau))) = \underline{f}$$

como buscábamos.

Por último, si $(\mathfrak{D}, \underline{\sigma}, \rho)(\langle S\rangle\varphi) = \perp$ entonces $(\mathfrak{D}, \underline{\sigma}', \rho)(\varphi) = \perp$, siendo $\underline{\sigma}' = (\mathfrak{D}, \underline{\sigma}, \rho)(S)$. Por hipótesis de inducción tendremos entonces que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau](\varphi'(\sigma',\ \tau)) = \perp$$

con lo que resulta de forma inmediata que:

$$(\mathfrak{D}, \rho)[\underline{\sigma}/\sigma,\ id_V^{\mathfrak{D}}/\tau](\exists\sigma' \in^{est} (\sigma' \simeq SIG_S(\sigma) \wedge \Delta\ SIG_S(\sigma) \wedge \varphi'(\sigma',\ \tau))) = \perp$$

como queríamos demostrar. ■

6.- TRABAJOS RELACIONADOS.

El tratamiento, en lógicas de programas, de la semántica de un lenguaje de programación imperativo con las características que tiene el que presentamos aquí, ha sido llevado a cabo, generalmente, desde un punto de vista operacional. Ver, por ejemplo, como casos típicos, los de [Apt 81] y [Bak 80]. En estos enfoques, la llamada a un procedimiento declarado de la forma $P(\overline{t}\triangleright\overline{y})$ se interpreta mediante lo que se llama *aproximación sintáctica*. Según ella, los parámetros actuales por valor son cedidos a una tupla de variables locales, pasando a ejecutarse el cuerpo de la declaración, sustituyendo los formales por referencia, por sus actuales, y los formales por valor, por las variables locales.

Esta forma de interpretar los programas no es muy conforme, como afirma [Bak 80], con los principios generales de la semántica denotacional. Además exige, como dijimos, un tratamiento riguroso del concepto de sustitución.

La semántica denotacional para lenguajes de programación, ya tratada en trabajos como [Ten 76] o el citado [Sto 77], ha sido la base de distintos estudios sobre su aplicación para lógicas de programas. Cabe citar entre ellos, los siguientes.

El trabajo de Reynolds [Rey 81] trata estos mismos problemas en su llamada *specification logic*, un sistema de corrección de programas que admite procedimientos con paso de parámetros por nombre. Este mecanismo consiste en sustituir el parámetro formal por el actual, literalmente, esto es, independientemente de que el primero esté ligado; una vez sustituido, el parámetro actual recibe su valor. Con él pueden ser simulados el paso de parámetros por valor y por referencia.

El trabajo de Trakhtenbrot et al. [THM 83] discute las ventajas de la semántica denotacional sobre la operacional, para un lenguaje de programación con las características del presentado en este capítulo, pero admitiendo procedimientos de orden superior (finito). La semántica del lenguaje se presenta de una forma similar a la que hemos visto: traduciendo los programas a un λ-cálculo tipado con recursión y asignando significado a las expresiones así obtenidas. Se discute sobre una teoría de prueba de corrección parcial, al estilo de Hoare, aunque no se dan pruebas de su completitud.

En la línea de la especificación lógica de Reynolds se encuadra el trabajo de Sieber [Sie 85]. Se trata de una lógica de corrección parcial, al estilo de Hoare, para el razonamiento sobre los programas de un lenguaje

imperativo con mecanismos de paso de procedimientos como parámetros, tal como presentan lenguajes como el citado L_4 (cf. §3.14). La semántica del lenguaje es definida desde un punto de vista denotacional, sin traducción previa al λ-cálculo. Un cálculo de Hoare permitiendo cuantificación en las fórmulas de corrección parcial es presentado, aunque no se da ningún esquema de prueba de su completitud.

Por último, hacer referencia al trabajo de Goerdt [Goe 87], donde se ocupa de trasladar, al lenguaje L_4, los resultados que obtenía para el λ-cálculo (cf. §3.14), esto es, completitud relativa de una lógica de Hoare. Para ello define una traducción de los programas imperativos en el λ-cálculo -sobre la que nos hemos inspirado para definir la que se ha presentado aquí- con abstracción sobre variables de localización.

CONCLUSIONES

El objetivo de este trabajo ha sido el estudio de las funciones parciales dentro de diversos sistemas lógicos. Para ello nos hemos basado en dos hechos principales: la combinación del λ-cálculo con la lógica de predicados, para expresar propiedades de funciones definidas recursivamente, y el empleo de tres valores veritativos, como marco semántico para una lógica parcial.

Creemos haber mostrado de forma completa cuáles son las posibilidades óptimas de razonamiento dentro de una lógica con tres valores booleanos. Para ello hemos definido un concepto, lo suficientemente genérico, de relación de consecuencia y hemos estudiado las características y conexiones que presentan aquellas consecuencias lógicas que resultan de su desarrollo.

Como conclusión de todo ello, pensamos que se ha justificado suficientemente la elección de una relación de consecuencia para una lógica trivalorada.

Por otra parte, hemos obtenido una lógica de primer orden para el razonamiento sobre funciones parciales, con un operador sintáctico de definición. Para ello hemos mostrado una metodología, basada en el método de los tableaux, para estudiar su teoría de prueba y construir cálculos de secuencias correctos y completos.

A su vez, dicha lógica se ha extendido a una que permitiera el tratamiento de funciones recursivas parciales (de primer orden), para lo cual hemos seguido un enfoque funcional puro, basado en un λ-cálculo con operador de punto fijo.

Hemos mostrado que la lógica resultante puede ser usada para la especificación semántica de lenguajes de programación imperativos, con un carácter netamente denotacional, con las ventajas que ello lleva consigo en cuanto a claridad y simplicidad se refiere. Además, es posible llevar a cabo

una traducción, que preserva el valor semántico, desde una lógica de programas imperativos a una de programas funcionales.

Pensamos que es posible extender los resultados anteriores a una lógica que considere funciones recursivas parciales de orden superior, aunque creemos que deberíamos seguir restringiéndonos, como ocurre en los sistemas de corrección parcial, a los tipos de primer orden. A su vez, ello posibilitaría llevar a cabo una especificación semántica de lenguajes imperativos más complejos, con mecanismos de paso de procedimientos como parámetros.

APENDICES

APÉNDICE 1

1.-Definición. *Medida de complejidad.*

Dada una signatura de primer orden Σ, definimos la complejidad de un Σ-término t, escrito $|t|$, y una Σ-fórmula φ, escrito $|\varphi|$, por dos funciones $|-|:T_\Sigma \to \omega$ y $|-|:F_\Sigma \to \omega^2$, donde ω es el primer ordinal no finito y ω^2 es el supremo del conjunto de ordinales $\{\omega * n \ / \ n<\omega\}$ (cf. [Lev 79] para una definición formal de estos conceptos), definidas respectivamente por:

$|x| = |c| = 1$

$|f(t_1,\ldots,t_n)| = \text{máximo}\ \{|t_i| \ / \ 1 \le i \le n\} + 1$

$|\text{if } \chi \text{ then } t| = \text{máximo}\ \{|\chi|, |t|\} + 2$

$|\text{if } \chi \text{ then } t \text{ else } t'| = \text{máximo}\ \{|\chi|, |t|, |t'|\} + 2$

$|t=t'| = 2 * \text{máximo}\ \{|t|, |t'|\} + 1$

$|R(t_1,\ldots,t_n)| = 2 * \text{máximo}\ \{|t_i| \ / \ 1 \le i \le n\} + 1$

$|\Delta t| = 2 * |t|$

$|\neg\varphi| = |\varphi| + 1$

$$|\varphi \vee \psi| = \begin{cases} \text{máximo}\ \{|\varphi|, |\psi|\} + 1 & \text{si } \varphi \text{ y } \psi \text{ no tienen cuantificadores} \\ \omega + \text{máximo}\ \{|\varphi|, |\psi|\} & \text{en otro caso} \end{cases}$$

$|\Delta\varphi| = |\varphi| * 2$

$|\exists x\varphi| = \omega * 2 + |\varphi|$ ∎

La razón de definir la complejidad de ciertas fórmulas como un ordinal no finito es que $\neg\exists x\varphi$ debe ser más compleja que su constituyente $\Delta t \to \varphi[t/x]$, para cualquier término t; de esta forma, si la complejidad de t crece tanto como queramos, hemos de asegurarnos que nunca sobrepasemos la de $\neg\exists x\varphi$. Como la de t puede ser, en principio, cualquier número finito, entonces la de $\neg\exists x\varphi$ debe ser mayor que todas ellas, es decir, no finita.

Obsérvese en la definición anterior que para todo $t \in T_\Sigma$: $1 \le |t| < \omega$, para toda fórmula $\varphi \in F_\Sigma$ sin cuantificadores: $2 \le |\varphi| < \omega$, y para toda fórmula $\varphi \in F_\Sigma$ con alguna aparición de algún cuantificador: $\omega < |\varphi| < \omega^2$. Para esto último - lo primero es evidente - basta tener en cuenta que $\omega * k + n$, con $k, n \in \omega$, es un ordinal menor que ω^2 (cf. [Lev 79] sobre aritmética ordinal), y que la complejidad de cualquier fórmula que contenga algún cuantificador, es de este tipo, según demuestra el siguiente lema.

2.-Lema.

Si φ contiene algún cuantificador entonces $|\varphi| = \omega * k + k'$, para ciertos $k, k' \in \omega$.

Demostración. En efecto, por inducción sobre la estructura de φ. Los casos de base y el paso inductivo para el caso de la negación son obvios. Para la disyunción se tiene:

$|\varphi \vee \psi| = \omega + (\omega * k + k') = \omega * (k + 1) + k'$

pues $\phi * k$ no es más que la suma de k veces el ordinal ϕ.

Teniendo esto en cuenta:

$|\exists x \varphi| = \omega * 2 + |\varphi| = \omega * 2 + (\omega * k + k') = \omega * (k + 2) + k'$

Por último:

$|\Delta\varphi| = |\varphi| * 2 = (\omega * k + k') * 2 = (\omega * k + k') + (\omega * k + k') =$

$= \omega * k + (k' + \omega * k) + k'$ (ya que la suma de ordinales es asociativa)

$= \omega * k + \omega * k + k'$ (ya que $n + \omega = \omega$ luego $n + \omega + .. + \omega = \omega + .. + \omega$)

$= \omega * (2 * k) + k'$. ■

Obsérvese que si hubiéramos definido $|\Delta\varphi| = 2 * |\varphi|$ entonces:

$2 * (\omega * k + k') = 2 * (\omega \overset{k}{+ \ldots +} \omega + k') = 2 * \omega \overset{k}{+ \ldots +} 2 * \omega + k'$ (ya que el producto de ordinales es distributivo con respecto de la suma) =

$\omega \overset{k}{+ \ldots +} \omega + k'$ (pues $2 * \omega = \omega$) $= \omega * k + k'$

con lo cual las complejidades de $\Delta\varphi$ y φ serían iguales.

3.-Teorema.

Para la clasificación orientada hacia los tableaux dada tras §2.3.1, toda fórmula no básica tiene una complejidad mayor que la de sus constituyentes. Esto es:

(1) Para toda fórmula $\alpha \in$ ALFA: $|\alpha| > |\alpha_i|$, para todo i=1,..,m.

(2) Para toda fórmula $\beta \in$ BETA: $|\beta| > |\beta_i|$, para todo i=1,..,m.

(3) Para toda fórmula $\gamma \in$ GAMMA: $|\gamma| > |\gamma(t)|$, para todo $t \in T_\Sigma$.

(4) Para toda fórmula $\delta \in$ DELTA: $|\delta| > |\delta(t)|$, para todo $t \in T_\Sigma$.

Demostración. (1) Vayamos tomando uno por uno, cada caso dado tras la definición §2.3.1. Tenemos:

(α1) $|R(t_1,\ldots,t_n)| = 2 * \text{máximo}\ \{|t_i| \ / \ 1 \leq i \leq n\} + 1 > 2 * |t_i| = |\Delta t_i|$, para todo i=1,..,n.

(α2) Inmediato pues $|\neg R(t_1,\ldots,t_n)| = |R(t_1,\ldots,t_n)| + 1 > |R(t_1,\ldots,t_n)|$.

(α3) y (α4) son análogos, por ser la misma la definición para la igualdad.

(α5) $|\neg\neg\varphi| = |\varphi| + 2 > |\varphi|$.

(α6) $|\neg(\varphi\vee\psi)| = |\varphi\vee\psi| + 1$. Distinguimos dos casos:

caso 1: φ y ψ no tienen cuantificadores, entonces:

$|\varphi\vee\psi| + 1 =$ **máximo** $\{|\varphi|, |\psi|\} + 2 > |\varphi| + 1 (= |\neg\varphi|), |\psi| + 1$

caso 2: φ o ψ tienen algún cuantificador, entonces:

$|\varphi\vee\psi| + 1 = \omega +$ **máximo** $\{|\varphi|, |\psi|\} + 1 >$ **máximo** $\{|\varphi|, |\psi|\} + 1$

(ya que $\omega + \phi > \phi$, para todo ordinal $\phi = \omega * k + k'$)

$\geq |\varphi| + 1 (= |\neg\varphi|), |\psi| + 1$

(α7) $|\Delta f(t_1,\ldots,t_n)| = 2 *$ (**máximo** $\{|t_i| / 1\leq i\leq n\} + 1) =$

$= 2 *$ **máximo** $\{|t_i| / 1\leq i\leq n\} + 2 > 2 * |t_i| = |\Delta t_i|$,

para todo $i=1,\ldots,n$.

(α8) $|\Delta$**if** χ **then** $t| = 2 *$ **máximo** $\{|\chi|, |t|\} + 4$

$\geq 2 * |\chi| + 4 > 2 * |\chi| > |\chi|$ (recuérdese que χ no tiene cuantificadores)

$\geq 2 * |t| + 4 > 2 * |t| = |\Delta t|$

(α9) $|\neg\Delta\exists x\varphi|$. Distinguimos dos casos:

caso 1: φ no tiene cuantificadores, de manera que $|\varphi| = a$, donde $2\leq a<\omega$. Entonces:

$|\neg\Delta\exists x\varphi| = (\omega * 2 + a) * 2 + 1 = \omega * 4 + a + 1$

$|\exists x\neg\Delta\varphi| = \omega * 2 + a * 2 + 1$

$|\neg\exists x\neg(\neg\Delta\varphi\vee\neg\varphi)| = \omega * 2 +$ (**máximo** $\{|\neg\Delta\varphi|, |\neg\varphi|\} + 2) + 1 =$

$= \omega * 2 +$ (**máximo** $\{a * 2 + 1, a + 1\} + 2) + 1 =$

$= \omega * 2 + (a * 2 + 1) + 1$ (pues $2\leq a<\omega$) $=$

$= \omega * 2 + a * 2 + 2$

En ambos casos se tiene lo buscado pues, como hemos dicho, $\omega * n$ es la suma de n veces ω, y $\omega + \phi > \phi$, para cualquier $\phi = \omega * k + k'$.

caso 2: φ tiene algún cuantificador, de manera que $\omega<|\varphi|<\omega^2$. Según el lema 2 podemos escribir $|\varphi| = \omega * k + k'$, para ciertos k, $k'<\omega$. Entonces:

$|\neg\Delta\exists x\varphi| = (\omega * 2 + (\omega * k + k')) * 2 + 1 =$

$= \omega * (4 + 2 * k) + k' + 1$

$|\exists x\neg\Delta\varphi| = \omega * 2 + (\omega * k + k') * 2 + 1 =$

$= \omega * (2 + 2 * k) + k' + 1$

$|\neg\exists x\neg(\neg\Delta\varphi\vee\neg\varphi)| = (\omega * 2 + (\omega +$ **máximo** $\{|\neg\Delta\varphi|, |\neg\varphi|\}) + 1) + 1$

$= \omega * 2 + (\omega +$ **máximo** $\{\omega * (2 * k) + k' + 1,$

$\omega * k + k' + 1\} + 1) + 1 =$

$= \omega * 2 + (\omega + \omega * (2 * k) + k' + 2) + 1 =$

$= \omega * (3 + 2 * k) + k' + 3$

y se tiene lo dado. Obsérvese que la complejidad primera es mayor que la segunda por "dos ω" y que la tercera, por "una ω"; el 3 es despreciable en esta comparación pues $\omega * (3 + 2 * k) + k' + 3$ es menor que el supremo de $\omega * (3 + 2 * k) + k' + n$, para $n < \omega$, y éste a su vez es $\omega * (4 + 2 * k) + k'$, menor que $\omega * (4 + 2 * k) + + k' + 1$.

(α10) $|\neg\Delta\neg\varphi|$. De nuevo distinguimos dos casos:

caso 1: φ no tiene cuantificadores, entonces:

$$|\neg\Delta\neg\varphi| = 2 * |\varphi| + 3 > 2 * |\varphi| + 1 = |\neg\Delta\varphi|$$

caso 2: φ tiene algún cuantificador. Por el lema 2, supongamos que $|\varphi|$ es de la forma $\omega * k + k'$, entonces:

$$|\neg\Delta\neg\varphi| = (\omega * k + k' + 1) * 2 + 1 = \omega * 2 * k + k' + 2$$
$$> \omega * 2 * k + k' + 1 = |\neg\Delta\varphi|$$

(α11) $|\neg\Delta\Delta t| = 4 * |t| + 1 > 3$ (pues $|t| \geq 1$, para todo t) $= |\neg\Delta x_0| = F$

(α12) $|\neg\Delta\Delta\varphi| = |\varphi| * 4 + 1$ (pues el producto de ordinales es asociativo)
> 3 (pues $|\varphi| \geq 2$, para toda φ) $= F$

(α13) $|\Delta\Delta t| = 4 * |t| > 2 = |\Delta x_0| = T$ y análogamente (α14).

(2) De nuevo tomamos los casos de §2.3.1. Tendremos entonces:

(β1) es inmediata si no aparecen cuantificadores. En otro caso:

$|\varphi\vee\psi| = \omega + \text{máximo}\ \{|\varphi|, |\psi|\} > \text{máximo}\ \{|\varphi|, |\psi|\}$, pues $\omega + \phi > \phi$, para cualquier $\phi = \omega * k + k'$.

(β2) $|\Delta \text{if } \chi \text{ then t else t'}| = 2 * \text{máximo}\ \{|\chi|, |t|, |t'|\} + 4$

$|\chi\wedge\Delta t| = |\neg(\neg\chi\vee\neg\Delta t)| = \text{máximo}\ \{|\chi| + 1, 2 * |t| + 1\} + 1$ (pues χ no tiene cuantificadores)

$\leq \text{máximo}\ \{2 * |\chi|, 2 * |t| + 1\} + 1$ (pues $|\chi| \geq 2$)

$< \text{máximo}\ \{2 * |\chi| + 4, 2 * |t| + 4\}$

$|\neg\chi\wedge\Delta t'| \leq \text{máximo}\ \{2 * |\chi| + 1, 2 * |t'| + 1\} + 1$ (análogamente)

$< \text{máximo}\ \{2 * |\chi| + 4, 2 * |t'| + 4\}$

y se tiene de forma inmediata lo buscado.

(β3) $|\neg\Delta \text{if } \chi \text{ then t else t'}| = 2 * \text{máximo}\ \{|\chi|, |t|, |t'|\} + 5$

$|\chi\wedge\neg\Delta t| = |\neg(\neg\chi\vee\neg\neg\Delta t)| = \text{máximo}\ \{|\chi| + 1, 2 * |t| + 2\} + 1$

$< \text{máximo}\ \{2 * |\chi| + 5, 2 * |t| + 5\}$ (como en (β2))

$|\neg\chi\wedge\neg\Delta t'| \leq \text{máximo}\ \{2 * |\chi| + 1, 2 * |t'| + 2\} + 1$

$< \text{máximo}\ \{2 * |\chi| + 5, 2 * |t'| + 5\}$ (como en (β2))

$|\neg\Delta\chi| = 2 * |\chi| + 1 < 2 * |\chi| + 5$

y el objetivo es inmediato.

(β4) $|\neg\Delta$if χ then t$| = 2 * $ máximo $\{|\chi|, |t|\} + 5$

$|\chi\wedge\neg\Delta t| = |\neg(\neg\chi\vee\neg\neg\Delta t)| =$ máximo $\{|\chi| + 1, 2 * |t| + 2\} + 1$

$<$ máximo $\{2 * |\chi| + 5, 2 * |t| + 5\}$

$|\neg\chi| = |\chi| + 1 < 2 * |\chi| + 5$

$|\neg\Delta\chi| = 2 * |\chi| + 1 < 2 * |\chi| + 5$

y de nuevo el objetivo es inmediato.

(β5) $|\neg\Delta(\varphi\vee\psi)|$. Distinguiremos casos como es habitual. En ellos solo comprobaremos que la complejidad de $\neg(\Delta\varphi\vee\Delta\psi)$ es menor, pues los otros dos constituyentes tienen complejidad aún menor que éste. Entonces:

caso 1: φ y ψ no tienen cuantificadores con lo cual:

$|\neg\Delta(\varphi\vee\psi)| =$ (máximo $\{|\varphi|, |\psi|\} + 1) * 2 + 1$

$=$ máximo $\{|\varphi| * 2, |\psi| * 2\} + 3$

$|\neg(\Delta\varphi\vee\Delta\psi)| =$ (máximo $\{|\varphi| * 2, |\psi| * 2\} + 1) + 1$

$=$ máximo $\{|\varphi| * 2, |\psi| * 2\} + 2$

caso 2: φ o ψ contienen algún cuantificador. Supongamos, sin pérdida de generalidad, que $|\varphi| \geq |\psi|$ y $|\varphi| = \omega * k + k'$, entonces:

$|\neg\Delta(\varphi\vee\psi)| = (\omega +$ máximo $\{|\varphi|, |\psi|\}) * 2 + 1$

$= (\omega + (\omega * k + k')) * 2 + 1$

$= (\omega * (k + 1) + k') * 2 + 1$

$= \omega * (2 * k + 2) + k' + 1$

$|\neg(\Delta\varphi\vee\Delta\psi)| = (\omega +$ máximo $\{|\varphi| * 2, |\psi| * 2\}) + 1$

$= \omega + (\omega * k + k') * 2 + 1$

$= \omega * (2 * k + 1) + k' + 1$

y en ambos casos el objetivo es obvio.

(β6) $|\neg\Delta t=t'| = 4 *$ máximo $\{|t|, |t'|\} + 3 > 2 * |t| + 1 = |\neg\Delta t|$ y análogamente con t'.

(β7) $|\Delta\varphi| = |\varphi| * 2 > |\varphi|, |\neg\varphi|$ $(= |\varphi| + 1)$, pues $|\varphi| \geq 2$.

Para demostrar (3) y (4) estableceremos en primer lugar los hechos siguientes:

(i) $|t[s/x]| = |t| + n$

(ii) $|\varphi[s/x]| = |\varphi| + n$

para algún $n < \omega$.

(i) es evidente si tenemos en cuenta que $|t|$ es siempre un número finito con lo que $n = |t[s/x]| - |t|$, pues obviamente $|t[s/x]|$ es siempre mayor o igual que $|t|$.

(ii) Cabe decir lo mismo, si φ no contiene cuantificadores con lo que supondremos, en adelante, que sí los tiene. Por tanto, según el lema 2 podemos escribir: $|\varphi| = \omega * k + k'$. Entonces aplicando inducción sobre la estructura de φ se tiene que:

($\neg$) $|(\neg\varphi)[s/x]| = |\neg\varphi[s/x]| = |\varphi[s/x]| + 1 = |\varphi| + n + 1$ (por HI) (léase "por hipótesis de inducción")

$= |\neg\varphi| + n$

(Δ) $|(\Delta\varphi)[s/x]| = |\Delta\varphi[s/x]| = |\varphi[s/x]| * 2 = ((\omega * k + k') + n) * 2$ (por HI) $= \omega * (2 * k) + k' + n = (\omega * k + k') * 2 + n = |\Delta\varphi| + n$

(v) Por hipótesis, $\psi \vee \psi'$ contienen cuantificadores. Distinguimos los dos casos siguientes:

caso 1: Solo una de las dos fórmulas de la disyunción contiene cuantificadores. Supongamos, sin pérdida de generalidad, que es ψ, con lo cual $|\psi'| < |\psi|$. Usando la hipótesis de inducción se tiene que $|\psi'[s/x]| < |\psi[s/x]|$. Entonces:

$$|(\psi \vee \psi')[s/x]| = |\psi[s/x] \vee \psi'[s/x]| = \omega + |\psi[s/x]| =$$
$$= \omega + |\psi| + n = \omega + \text{máximo } \{|\psi|, |\psi'|\} + n = |\psi \vee \psi'| + n$$

caso 2: Las dos fórmulas contienen cuantificadores. Escribamos entonces: $|\psi| = \omega * k + k'$, $|\psi[s/x]| = |\psi| + n$, $|\psi'| = \omega * l + l'$ y $|\psi'[s/x]| = |\psi'| + n'$.

Si $k > l$ entonces $|\psi| > |\psi'|$ y $|\psi[s/x]| > |\psi'[s/x]|$, y estamos como en el caso 1. Si $k < l$, estaremos en un caso simétrico. Supongamos pues que $k = l$. Si $k' \geq l'$ y $k' + n \geq l' + n'$ (o, simétricamente, si $k' < l'$ y $k' + n < l' + n'$), es evidente que basta seguir de nuevo el caso anterior. Por tanto supongamos, sin pérdida de generalidad, uno de los dos casos que restan, es decir:

$k = l$, $k' \geq l'$, $k' + n < l' + n'$

entonces:

$$|(\psi \vee \psi')[s/x]| = \omega + \text{máximo } \{|\psi[s/x]|, |\psi'[s/x]|\} =$$
$$= \omega + \text{máximo } \{\omega * k + k' + n, \omega * k + l' + n'\} =$$
$$= \omega + \omega * k + l' + n'$$
$$|\psi \vee \psi'| = \omega + \text{máximo } \{\omega * k + k', \omega * k + l'\} =$$
$$= \omega + \omega * k + k'$$

con lo cual $|(\psi \vee \psi')[s/x]| = |\psi \vee \psi'| + (l' + n' - k')$.

(3) Para $\exists y\psi$, si $y=x$, es trivial tomando $n = 0$. Si y aparece en s entonces $(\exists y\psi)[s/x] = \exists u(\psi[u/y])[s/x]$, siendo u nueva con respecto a ψ. Pero en este caso es evidente que $|\psi[u/y]| = |\psi|$, ya que se trata de

una sustitución literal de una variable por otra, de manera que $|\exists u\psi[u/y]| = |\exists u\psi'|$ y podemos aplicar el caso general que vemos a continuación, en el que tenemos:

$$|(\exists y\psi)[s/x]| = |\exists y\psi[s/x]| = \omega * 2 + |\psi[s/x]| = \omega * 2 + (|\psi| + n)$$

(por HI) $= (\omega * 2 + |\psi|) + n = |\exists y\psi| + n.$

Demostrados los hechos (i) y (ii), pasamos a demostrar (3) y (4) del teorema. Para (3), si φ no tiene cuantificadores entonces es trivial pues $|\Delta t \rightarrow \varphi[t/x]|$ es finita y la de $|\neg\exists x\varphi|$ no lo es. Supongamos pues que φ contiene cuantificadores. Por (ii) tenemos:

$|\Delta t \rightarrow \varphi[t/x]| = |\neg\Delta t \vee \varphi[t/x]| = \omega + |\varphi[t/x]|$ (pues $|\neg\Delta t|$ es un número finito y $|\varphi[t/x]|$, no) $= \omega + |\varphi| + n$

para algún $n < \omega$. Por otra parte:

$$|\neg\exists x\varphi| = \omega * 2 + |\varphi| + 1$$

y vale lo dicho en el caso 2 de (α9) para demostrar que n es despreciable frente al ω que $\neg\exists x\varphi$ saca a su constituyente $\Delta t \rightarrow \varphi[t/x]$.

Por último, para demostrar (4) tenemos que:

$$|\exists x\varphi| = \omega * 2 + |\varphi|$$

$$|\varphi[t/x]| = |\varphi| + n$$

y valen los mismos comentarios que acabamos de hacer. ■

APÉNDICE 2

El propósito de este apéndice es demostrar que para todo axioma de la igualdad θ se tiene que $\vdash \theta$ y $\vdash \Delta\theta$. Antes de verlo, demostraremos algunos esquemas de reglas sobre la definición de conectivas, cuantificadores e igualdad con constantes auxiliares, que serán frecuentemente usados en la prueba.

$(D\wedge)$ $\dfrac{\Gamma,\ \psi\ \ \Delta\varphi,\quad \Gamma,\ \varphi\ \ \Delta\psi,\quad \Gamma\ \ \Delta\varphi}{\Gamma\ \ \Delta(\varphi\wedge\psi)}$ $\qquad$ $\dfrac{\Gamma,\ \psi\ \ \Delta\varphi,\quad \Gamma,\ \varphi\ \ \Delta\psi,\quad \Gamma\ \ \Delta\psi}{\Gamma\ \ \Delta(\varphi\wedge\psi)}$

1.	$\Gamma,\ \psi\ \ \Delta\varphi$	Hipótesis
2.	$\Gamma,\ \varphi\ \ \Delta\psi$	Hipótesis
3.	$\Gamma\ \ \Delta\varphi$	Hipótesis
4.	$\Gamma,\ \psi\ \ \Delta\neg\varphi$	$(D\neg)$, 1
5.	$\Gamma\ \ \Delta\neg\varphi\vee\neg\psi$	$(I\rightarrow C2)$, 4
6.	$\Gamma\ \ \Delta\neg\psi\vee\neg\varphi$	Análogo a 4, 5, para 2
7.	$\Gamma\ \ \Delta\neg\varphi$	$(D\neg)$, 3
8.	$\Gamma\ \ \Delta\neg\varphi\vee\Delta\neg\psi$	(IvC), 7
9.	$\Gamma\ \ \Delta(\neg\varphi\vee\neg\psi)$	(Dv), 5, 6, 8
10.	$\Gamma\ \ \Delta\neg(\neg\varphi\vee\neg\psi)$	$(D\neg)$, 9
11.	$\Gamma\ \ \Delta(\varphi\wedge\psi)$	Definición de $\wedge$, 10

$(D\rightarrow)$ $\dfrac{\Gamma\ \ \Delta\varphi,\quad \Gamma,\ \varphi\ \ \Delta\psi}{\Gamma\ \ \Delta(\varphi\rightarrow\psi)}$

1.	$\Gamma\ \ \Delta\varphi$	Hipótesis
2.	$\Gamma,\ \varphi\ \ \Delta\psi$	Hipótesis
3.	$\Gamma\ \ \Delta\neg\varphi$	$(D\neg)$, 1
4.	$\Gamma\ \ \Delta\neg\varphi\vee\psi$	(IvC), 3
5.	$\Gamma\ \ \Delta\psi\vee\neg\varphi$	$(I\rightarrow C2)$, 2
6.	$\Gamma\ \ \Delta\neg\varphi\vee\Delta\psi$	(IvC), 3
7.	$\Gamma\ \ \Delta(\neg\varphi\vee\psi)$	(Dv), 4, 5, 6
8.	$\Gamma\ \ \Delta(\varphi\rightarrow\psi)$	Definición de $\rightarrow$, 7

(D∀) $\dfrac{\Gamma \ \Delta\varphi[c/x]}{\Gamma \ \Delta\forall x\varphi}$ donde $c\in C$ no aparece en $\Gamma \cup \{\varphi\}$

1. $\Gamma \ \Delta\varphi[c/x]$ Hipótesis
2. $\Gamma \ \Delta\neg\varphi[c/x]$ (D¬), 1
3. $\Gamma \ \neg\neg\Delta\neg\varphi[c/x]$ (I¬¬C), 2
4. $\Gamma \ \neg\exists x\neg\Delta\neg\varphi$ (I∀C), 3
5. $\Gamma \ \Delta\exists x\neg\varphi$ (D∃), 4
6. $\Gamma \ \Delta\neg\exists x\neg\varphi$ (D¬), 5
7. $\Gamma \ \Delta\forall x\varphi$ Definición de ∀, 6

(DIgCon) $\Delta c\approx c'$ $\Delta\neg(c\approx c')$, donde $c, c'\in C$

1. $\neg\Delta c \ \ F$ (Ctd)
2. Δc (RFT), 1
3. $\Delta c'$ Análogo a 1, 2
4. $\Delta c\approx c'$ (DIg), 2, 3
5. $\Delta\neg(c\approx c')$ (D¬), 4

1.-Teorema.

Para todo axioma de la igualdad θ se tiene: $\vdash \theta$ y $\vdash \Delta\theta$.

Demostración. Distinguiremos el tipo de axioma lógico de que se trate, mostrando las dos derivaciones. Se tiene entonces:

(1) $\forall x(x\approx x)$

1. $c\approx c$ (REF)
2. $\forall x(x\approx x)$ (I∀C2), 1

1. $\Delta c\approx c$ (DIgCon)
2. $\Delta\forall x(x\approx x)$ (D∀), 1

(2) $\forall x\forall y(x\approx y\rightarrow y\approx x)$

1. $c\approx c$ (REF)
2. $c\approx c' \ \ c'\approx c$ (Subs), 1, $c\approx c'$
3. $c\approx c'\rightarrow c'\approx c$ (I→C), 2
4. $\forall x\forall y(x\approx y\rightarrow y\approx x)$ (I∀C2), 3, 2 veces

1. $\Delta c \approx c'$ (DIgCon), $c \neq c'$
2. $\Delta c' \approx c$ (DIgCon)
3. $c \approx c'$ $\Delta c' \approx c$ (IHip), 2
4. $\Delta(c \approx c' \rightarrow c' \approx c)$ (D$\rightarrow$), 1, 3
5. $\Delta \forall x \forall y(x \approx y \rightarrow y \approx x)$ (D$\forall$), 4, 2 veces

(3) $\forall x \forall y \forall z(x \approx y \wedge y \approx z \rightarrow x \approx z)$

1. $c \approx c'$ $c \approx c'$ (Hip), $c \neq c'$
2. $c \approx c'$, $c' \approx c''$ $c \approx c''$ (Subs), 1, c, $c' \neq c''$
3. $c \approx c' \wedge c' \approx c''$ $c \approx c''$ (I$\wedge$A), 2
4. $c \approx c' \wedge c' \approx c'' \rightarrow c \approx c''$ (I$\rightarrow$C), 3
5. $\forall x \forall y \forall z(x \approx y \wedge y \approx z \rightarrow x \approx z)$ (I$\forall$C2), 4, 3 veces

1. $\Delta c \approx c'$ (DIgCon), $c \neq c'$
2. $\Delta c' \approx c''$ (DIgCon), $c' \neq c''$
3. $\Delta c \approx c''$ (DIgCon), $c \neq c''$
4. $c' \approx c''$ $\Delta c \approx c'$ (IHip), 1
5. $c \approx c'$ $\Delta c' \approx c''$ (IHip), 2
6. $\Delta(c \approx c' \wedge c' \approx c'')$ (D$\wedge$), 4, 5, 1
7. $c \approx c' \wedge c' \approx c''$ $\Delta c \approx c''$ (IHip), 3
8. $\Delta(c \approx c' \wedge c' \approx c'' \rightarrow c \approx c'')$ (D$\rightarrow$), 6, 7
9. $\Delta \forall x \forall y \forall z(x \approx y \wedge y \approx z \rightarrow x \approx z)$ (D$\forall$), 8, 3 veces

(4) Supongamos, para simplificar la notación, que f es unaria. Entonces para $\forall x \forall y(x \approx y \wedge \Delta f(x) \rightarrow \Delta f(y))$ se tiene:

1. $\Delta f(c)$ $\Delta f(c)$ (Hip)
2. $c \approx c'$, $\Delta f(c)$ $\Delta f(c')$ (Subs), 1, $c \neq c'$
3. $c \approx c' \wedge \Delta f(c)$ $\Delta f(c')$ (I$\wedge$A), 2
4. $c \approx c' \wedge \Delta f(c) \rightarrow \Delta f(c')$ (I$\rightarrow$C), 3
5. $\forall x \forall y(x \approx y \wedge \Delta f(x) \rightarrow \Delta f(y))$ (I$\forall$C2), 4, 2 veces

1. $\Delta c \approx c'$ (DIgCon), $c \neq c'$
2. $\Delta f(c)$ $\Delta c \approx c'$ (IHip), 1
3. $\Delta \Delta f(c)$ (DFor)
4. $c \approx c'$ $\Delta \Delta f(c)$ (IHip), 3
5. $\Delta(c \approx c' \wedge \Delta f(c))$ (D$\wedge$), 2, 4, 1
6. $\Delta \Delta f(c')$ (DFor)

7. $c{=}c' \wedge \Delta f(c) \ \ \Delta\Delta f(c')$ (IHip), 6
8. $\Delta(c{=}c' \wedge \Delta f(c) \rightarrow \Delta f(c'))$ (D$\rightarrow$), 5, 7
9. $\Delta\forall x\forall y(x{=}y \wedge \Delta f(x) \rightarrow \Delta f(y))$ (D$\forall$), 8, 2 veces

(5) De nuevo suponemos, para simplificar la notación, que χ solo tiene una variable libre. Entonces para $\forall x\forall y(x{=}y \wedge \Delta\chi(x) \wedge \chi(x) \rightarrow \chi(y))$ se tiene:

1. $\Delta\chi(c), \ \chi(c) \ \ \chi(c)$ (Hip)
2. $c{=}c', \ \Delta\chi(c), \ \chi(c) \ \ \chi(c')$ (Subs), 1, $c{=}c'$
3. $c{=}c' \wedge \Delta\chi(c) \wedge \chi(c) \ \ \chi(c')$ (I$\wedge$A), 2, 2 veces
4. $c{=}c' \wedge \Delta\chi(c) \wedge \chi(c) \rightarrow \chi(c')$ (I$\rightarrow$C), 3
5. $\forall x\forall y(x{=}y \wedge \Delta\chi(x) \wedge \chi(x) \rightarrow \chi(y))$ (I$\forall$C2), 4, 2 veces

1. $\Delta\Delta\chi(c)$ (DFor)
2. $\chi(c) \ \ \Delta\Delta\chi(c)$ (IHip), 1
3. $\Delta\chi(c) \ \ \Delta\chi(c)$ (Hip)
4. $\Delta(\Delta\chi(c) \wedge \chi(c))$ (D$\wedge$), 2, 3, 1
5. $\Delta c{=}c'$ (DIgCon)
6. $\Delta\chi(c) \wedge \chi(c) \ \ \Delta c{=}c'$ (IHip), 5
7. $c{=}c' \ \ \Delta(\Delta\chi(c) \wedge \chi(c))$ (IHip), 4
8. $\Delta(c{=}c' \wedge \Delta\chi(c) \wedge \chi(c))$ (D$\wedge$), 6, 7, 5
9. $\Delta\chi(c), \ \chi(c) \ \ \Delta\chi(c)$ (Hip)
10. $c{=}c', \ \Delta\chi(c), \ \chi(c) \ \ \Delta\chi(c')$ (Subs), 9, $c{=}c'$
11. $c{=}c' \wedge \Delta\chi(c) \wedge \chi(c) \ \ \Delta\chi(c')$ (I$\wedge$A), 10, 2 veces
12. $\Delta(c{=}c' \wedge \Delta\chi(c) \wedge \chi(c) \rightarrow \chi(c'))$ (D$\rightarrow$), 8, 11
13. $\Delta\forall x\forall y(x{=}y \wedge \Delta\chi(x) \wedge \chi(x) \rightarrow \chi(y))$ (D$\forall$), 12, 2 veces

(6) $\forall x(\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } x){=}x)$

1. $(\text{if } \chi \text{ then } c){=}c$ (IF), $c \in C$ no aparece en χ
2. $\chi, \ \Delta\chi \ \ (\text{if } \chi \text{ then } c){=}c$ (IHip), 1
3. $\chi \wedge \Delta\chi \ \ (\text{if } \chi \text{ then } c){=}c$ (I$\wedge$A), 2
4. $\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } c){=}c$ (I$\rightarrow$C), 3
5. $\forall x(\chi \wedge \Delta\chi \rightarrow (\text{if } \chi \text{ then } x){=}x)$ (I$\forall$C2), 4

1. $\Delta\Delta\chi$ (DFor)
2. $\Delta\chi \ \ \Delta\chi$ (Hip), 1
3. $\chi \ \ \Delta\Delta\chi$ (IHip), 1
4. $\Delta(\chi \wedge \Delta\chi)$ (D$\wedge$), 2, 3, 1

5. $\neg\Delta c$ F (Ctd), $c \in C$ no aparece en χ
6. Δc (RFT), 5
7. $\chi \wedge \Delta\chi$ Δc (IHip), 6
8. χ, $\Delta\chi$ χ (Hip)
9. $\chi \wedge \Delta\chi$ χ (I$\wedge$A), 8
10. χ, $\Delta\chi$ $\Delta\chi$ (Hip)
11. $\chi \wedge \Delta\chi$ $\Delta\chi$ (I$\wedge$A), 10
12. $\chi \wedge \Delta\chi$ Δif χ then c (DIF), 9, 11, 7
13. $\chi \wedge \Delta\chi$ Δ(if χ then c)$\approx$c (DIg), 10, 7
14. $\Delta(\chi \wedge \Delta\chi \rightarrow$(if χ then c)$\approx$c) (D$\rightarrow$), 4, 13
15. $\Delta(\forall x(\chi \wedge \Delta\chi \rightarrow$(if χ then x)$\approx$x)) (D$\forall$), 14

(7) $[\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t$] \wedge$
$[\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t'$]$
1. χ (if χ then t else t')$\approx$t (IFT)
2. χ, $\Delta\chi$, Δt (if χ then t else t')$\approx$t (IHip), 1
3. $\chi \wedge \Delta\chi \wedge \Delta$t (if χ then t else t')$\approx$t (I$\wedge$A), 2, 2 veces
4. $\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t (I$\rightarrow$C), 3
5. $\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t' Análogo a 1-4, con (IFE)
6. $[\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t$] \wedge$
$\wedge$ $[\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t'$]$ (I$\wedge$C2), 4, 5

1. $\Delta(\chi \wedge \Delta\chi)$ Como en (6), 1-4
2. Δt $\Delta(\chi \wedge \Delta\chi)$ (IHip), 1
3. $\Delta\Delta t$ (DTer)
4. $\chi \wedge \Delta\chi$ $\Delta\Delta t$ (IHip), 3
5. $\Delta(\chi \wedge \Delta\chi \wedge \Delta t)$ (D$\wedge$), 2, 4, 1
6. χ, $\neg\chi$ $\Delta t'$ (Ctd)
7. χ, $\Delta\chi$, Δt, $\neg\chi$ $\Delta t'$ (IHip), 6
8. χ, $\Delta\chi$, Δt $\neg\chi \rightarrow \Delta t'$ (I$\rightarrow$C), 7
9. $\chi \wedge \Delta\chi \wedge \Delta t$ $\neg\chi \rightarrow \Delta t'$ (I$\wedge$A), 8, 2 veces
10. χ, $\Delta\chi$, Δt Δt (Hip)
11. $\Delta\chi$, Δt $\chi \rightarrow \Delta t$ (I$\rightarrow$C), 10
12. χ, $\Delta\chi$, Δt $\chi \rightarrow \Delta t$ (IHip), 11
13. $\chi \wedge \Delta\chi \wedge \Delta t$ $\chi \rightarrow \Delta t$ (I$\wedge$A), 12, 2 veces
14. χ, $\Delta\chi$, Δt $\Delta\chi$ (Hip)
15. $\chi \wedge \Delta\chi \wedge \Delta t$ $\Delta\chi$ (I$\wedge$A), 14, 2 veces

16. $\chi \wedge \Delta\chi \wedge \Delta t$ Δif χ then t else t' (DIFT), 15, 13, 9

17. $\chi \wedge \Delta\chi \wedge \Delta t$ Δt (I$\wedge$A), 10, 2 veces

18. $\chi \wedge \Delta\chi \wedge \Delta t$ Δ(if χ then t else t')$\approx$t (DIg), 16, 17

19. $\Delta[\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t] (D$\rightarrow$), 5, 18

20. $\Delta[\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t'] Análogo a 1-19

21. $\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t'
$\Delta[\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t] (IHip), 19

22. $\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t
$\Delta[\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t'] (IHip), 20

23. Δ([$\chi \wedge \Delta\chi \wedge \Delta t \rightarrow$(if χ then t else t')$\approx$t]$\wedge$
$\wedge[\neg\chi \wedge \Delta\chi \wedge \Delta t' \rightarrow$(if χ then t else t')$\approx$t']) (D$\wedge$), 21, 22, 19

■

APENDICE 3

El propósito de este apéndice es demostrar que para todo axioma de la igualdad θ se tiene $\vdash \theta$ y $\vdash \Delta\theta$, entendiendo que la derivabilidad se refiere ahora al cálculo de secuencias $\mathfrak{GJX}$ que se definió en el capítulo tercero. Para la demostración emplearemos las reglas sobre definición de las conectivas (D$\vee$), (D$\wedge$) y D($\rightarrow$), así como las que se refieren a la definición de los cuantificadores (D$\forall$) y de las constantes auxiliares (DIgCon) (cf. apéndice 2 para detalles).

1.-Teorema.

Para todo axioma de la igualdad θ se tiene: $\vdash \theta$ y $\vdash \Delta\theta$.

Demostración. Para aquellos axiomas que ya existían en el cálculo para LFP, vale la misma demostración. Para el axioma:

$$\forall\bar{x}\forall\bar{y}(\bar{x}\approx\bar{y}\wedge\Delta X(\bar{x}) \rightarrow \Delta X(\bar{y}))$$

la demostración es análoga a la dada para símbolos de función.

La demostración para el axioma sobre la construcción condicional $\chi\rightarrow E, E'$ es, cambiando términos por expresiones, la misma que la dada en el apéndice 2.

Nos ocuparemos, por tanto, solamente de los axiomas que se refieren a las definiciones por abstracción y a las definiciones recursivas. Para ellos se tiene:

(1) $\forall y_1 .. \forall y_n (\Delta E[y_1/x_1,...,y_n/x_n] \rightarrow$

$$\rightarrow (\lambda x_1 .. x_n . E)(y_1,..,y_n) \approx E[y_1/x_1,..,y_n/x_n])$$

Supongamos, sin pérdida de generalidad, que la función definida es de un solo argumento; las derivaciones del axioma y de su definición son, entonces, las siguientes, donde c es una constante auxiliar del mismo tipo que la variable que sustituye:

1. $(\lambda x.E)(c) \approx E[c/x]$	(ABS)
2. $\Delta E[c/x]$ $(\lambda x.E)(c) \approx E[c/x]$	(IHip), 1
3. $\Delta E[c/x] \rightarrow (\lambda x.E)(c) \approx E[c/x]$	(I$\rightarrow$C), 2
4. $\forall y(\Delta E[y/x] \rightarrow (\lambda x.E)(y) \approx E[y/x])$	(I$\forall$C2), 3

1. $\Delta\Delta E[c/x]$	(DE)
2. $\Delta E[c/x]$ $\Delta E[c/x]$	(Hip)
3. $\Delta E[c/x]$, Δc $\Delta E[c/x]$	(IHip), 2

4. $\Delta E[c/x],\ \Delta c\ \ \Delta c$ (Hip)
5. $\Delta E[c/x],\ \Delta c\ \ \Delta(\lambda x.E)(c)$ (DAbs), 3, 4
6. $\Delta E[c/x],\ \Delta c\ \ \Delta((\lambda x.E)(c) \simeq E[c/x])$ (DIg), 5, 3
7. $\Delta E[c/x]\ \ \Delta((\lambda x.E)(c) \simeq E[c/x])$ (DCon), 6
8. $\Delta(\Delta E[c/x] \rightarrow (\lambda x.E)(c) \simeq E[c/x])$ (D$\rightarrow$), 1, 7
9. $\Delta\forall y(\Delta E[y/x] \rightarrow (\lambda x.E)(y) \simeq E[y/x])$ (D$\forall$), 8

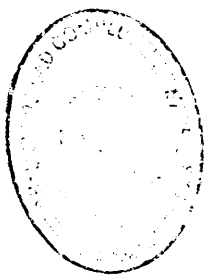
(2) $\forall\overline{x}(\Delta(\mu X.M)^i(\overline{x}) \rightarrow (\mu X.M)(\overline{x}) \simeq (\mu X.M)^i(\overline{x}))$ $(i<\omega)$

Como en el caso anterior, supongamos, sin pérdida de generalidad, que la función definida por recursión es de un solo argumento. Entonces, dado $i<\omega$ cualquiera, se tiene:

1. $(\mu X.M)(c) \simeq (\mu X.M)^i(c)$ (REC)
2. $\Delta(\mu X.M)^i(c)\ \ (\mu X.M)(c) \simeq (\mu X.M)^i(c)$ (IHip), 1
3. $\Delta(\mu X.M)^i(c) \rightarrow (\mu X.M)(c) \simeq (\mu X.M)^i(c)$ ($\underline{I}\rightarrow$C), 2
4. $\forall x(\Delta(\mu X.M)^i(x) \rightarrow (\mu X.M)(x) \simeq (\mu X.M)^i(x))$ (I$\forall$C2), 3

1. $\Delta\Delta(\mu X.M)^i(c)$ (DE)
2. $\Delta(\mu X.M)^i(c)\ \ \Delta(\mu X.M)^i(c)$ (Hip)
3. $\Delta(\mu X.M)^i(c)\ \ \Delta(\mu X.M)(c)$ (DRec), 2
4. $\Delta(\mu X.M)^i(c)\ \ \Delta((\mu X.M)(c) \simeq (\mu X.M)^i(c))$ (DIg), 2, 3
5. $\Delta(\Delta(\mu X.M)^i(c) \rightarrow ((\mu X.M)(c) \simeq (\mu X.M)^i(c)))$ (D$\rightarrow$), 1, 4
6. $\Delta\forall x(\Delta(\mu X.M)(x) \rightarrow ((\mu X.M)(x) \simeq (\mu X.M)^i(x)))$ (D$\forall$), 5

■



BIBLIOGRAFIA

[AN 80] *The metric space of infinite trees. Algebraic and topological properties.* A. Arnold, M. Nivat. Annales Societatis Mathematical Polonal. Series IV: Fundamenta Informaticae III 4, 445-476, 1980.

[Apt 81] *Ten years of Hoare's logic: a survey - Part I.* K. R. Apt. ACM Trans. Prog. Lang. Syst. 3, 431-483, 1981.

[Avr 87] *Simple consequence relations.* A. Avron. LFCS Report Series, ECS-LFCS-87-30, 1987.

[Avr 88] *Foundations and proof theory of 3-valued logics.* A. Avron. LFCS Report Series, ECS-LFCS-88-48, 1988.

[Bak 80] *Mathematical theory of program correctness.* J. de Bakker. Prentice-Hall, 1980.

[BCJ 84] *A logic covering undefinedness in program proofs.* H. Barringer, J. H. Cheng, C. B. Jones. Acta Informatica 21, 251-269, 1984.

[Ber 85] *Una exposición de los fundamentos matemáticos de la semántica denotacional.* J. Bermúdez. Tesina de Licenciatura. Facultad de Matemáticas, UCM, 1985.

[Bet 59] *The foundations of mathematics.* E. W. Beth. North-Holland, 1959.

[Bla 86] *Partial logic.* S. Blamey. Handbook of Philosophical Logic, Vol. III, 1-70, D. Reidel Publishing Company, 1986.

[Bli 88] *Three-valued predicates for software specification and validation.* A. Blikle. VDM'88, VDM-The way ahead. Proceedings. Lecture Notes in Computer Science 328, Springer-Verlag, 1988.

[Boc 39] *On a 3-valued logical calculus and its application to the analysis of contradictions.* D. A. Bochvar. Matématičeskij sbornik 4, 287-308, 1939.

[BW 82] *Partial abstract types.* M. Broy, M. Wirsing. Acta Informatica 18, 47-64, 1982.

[Cla 79] *Programming languages for which it is impossible to obtain good Hoare-like axioms.* E. M. Clarke, Journal of the ACM, vol. 26, 129-147, 1979.

[CK 73] *Model theory.* C. C. Chang, H. J. Keisler. North-Holland, 1973.

[Dal 80] *Logic and structure.* D. Van Dalen. Springer-Verlag, 1980.

[Del 88] *Programmation en logique trivaluée.* J. P. Delahaye. European Workshop on Logical Method in Artificial Intelligence, JELIA 88, 1988.

[DM 89] *Logique partielle et PROLOG.* J. P. Delahaye, P. Mathieu. Séminaire de Programmation Logique de CNET, 1989.

[Ebb 69] *Über eine prädikatenlogik mit partiell definierten prädikaten und funktionen.* H.-D. Ebbinghaus. Arch. Math. Logik 12, 39-53, 1969.

[EFT 84] *Mathematical logic.* H. D. Ebbinghaus, J. Flum, W. Thomas. Springer-Verlag, 1984.

[Fit 85] *A Kripke-Kleene semantics for logic programs.* M. Fitting. J. Logic Programming, 295-312, 1985.

[Fit 86] *Partial models and logic programming.* M. Fitting. Theoretical Computer Science 48, 229-255, 1986.

[GL 89] *A first order logic for partial functions.* A. Gavilanes, F. Lucio. Aparecerá en número extraordinario de Theoretical Computer Science.

[Goe 85] *A Hoare calculus for functions defined by recursion on higher type.* A. Goerdt. Logics of Programs. Proceedings. Lecture Notes in Computer Science 193, 106-117, 1985.

[Goe 86] *Ein Hoare kalkül für die sprache der getypten lambda-terme. Korrektheit, vollständigkeit, anwendungen.* A. Goerdt. Thesis, 1986.

[Goe 87] *Hoare logic for lambda-terms as basis of Hoare logic for imperative languages.* A. Goerdt. IEEE, 293-299, 1987.

[Har 79] *First-Order dynamic logic.* D. Harel. Lecture Notes in Computer Science, Springer-Verlag, 1979.

[Har 84] *Dynamic logic.* D. Harel. En: Gabbay and Guenthner (eds.), Handbook of Philosophical Logic, II: Extensions of Classical Logic, D. Reidel, Boston, 497-604, 1984.

[Hin 55] *Form and content in quantification theory.* J. Hintikka. Acta Phil. Fen. 8, 7-55, 1955.

[Hoo 83] *A partial-predicate calculus in a two-valued logic.* A. Hoogewijs. Zeitschrift f. Math. Logik u. Grundlagen d. Math., 29, 239-243, 1983.

[Hoo 87] *Partial-predicate logic in computer science.* A. Hoogewijs. Acta Informatica 24, 281-293, 1987.

[Kle 52] *Introduction to metamathematics.* S. C. Kleene. North-Holland, 1952.

[Kol 76] *Sequent calculus and partial logic.* G. Koletsos. MSc thesis, The University of Manchester, 1976.

[Kön 26] *Sur les correspondances multivoques des ensembles.* D. König. Fund. Math. 8, 114-134, 1926.

[KT] *Logics of programs*. D. Kozen, J. Tiuryn. Aparecerá como capítulo en: Handbook of Theoretical Computer Science. Managing Editor: J. v. Leeuwen. North-Holland.

[KTB 88] *A three-valued logic for software specification and validation*. B. Konikowska, A. Tarlecki, A. Blikle. VDM'88, VDM-The way ahead. Proceedings. Lecture Notes in Computer Science 328, Springer-Verlag, 1988.

[Lev 79] *Basic set theory*. A. Levy. Springer-Verlag, 1979.

[Loe 87] *Algorithmic specifications: a constructive specification method for abstract data types*. J. Loeckx. ACM Trans. on Prog. Lang., vol 9, n. 4, 646-685, 1987.

[Luk 20] *On 3-valued logic*. J. Lukasiewicz. En: McCall S. (ed.). Polish Logic, Oxford University Press, 1967.

[Man 74] *Mathematical theory of computation*. Z. Manna. McGraw Hill, 1974.

[McC 63] *A basis for a mathematical theory of computation*. J. McCarthy. Published in: Computer Programming and Formal Systems. North-Holland, 33-70, 1963.

[MM 83] *Termination assertions for recursive programs: Completeness and axiomatic definability*. A. R. Meyer, J. C. Mitchell. Information and Control 56, 112-138, 1983.

[Owe 84] *An approach to program reasoning based on a first-order logic for partial functions*. O. Owe. Computer Science Technical Report Num. CS-081. Dep. of Electrical Engineering and Computer Sciences. Univ. of California, San Diego, 1984.

[Pos 21] *Introduction to a general theory of elementary propositions*. E. L. Post. Amer. J. Math. 43, 163-185, 1921.

[Pro 85] *PROSPECTRA Project Summary*. B. Krieg-Brückner et al. Univ. Bremen, 1985.

[Rey 81] *The craft of programming*. J. C. Reynolds. Prentice Hall, 1981.

[Rog 67] *Theory of recursive functions and effective computability*. H. Rogers. McGraw Hill, 1967.

[Sco 70] *Outline of a mathematical theory of computation technical monograph PRG-2*. D. S. Scott. Oxford University Computing Laboratory, November 1970.

[Sco 76] *Data types as lattices*. D. S. Scott. SIAM Journal on Computing, Vol. 5, 522-587, 1976.

[Sco 82] *Domains for denotational semantics.* D. S. Scott. ICALP'82, Aarhus, Denmark, 1982.

[Sie 85] *A partial correctness logic for procedures.* K. Sieber. Logics of Programs. Proceedings. Lecture Notes in Computer Science 193, 320-342, 1985.

[Smu 68] *First-order logic.* R. M. Smullyan. Springer-Verlag, 1968.

[Sob 52] *Axiomatization of partial system of three-valued calculus of propositions.* B. Sobocinski. The Journal of Computing Systems, Vol. II, 23-55, 1952.

[Sto 77] *Denotational semantics: The Scott-Strachey approach to programming language theory.* J. Stoy. MIT Press, 1977.

[Ten 76] *The denotational semantics of programming languages.* R. D. Tennent. Comm. ACM 19 (8), 437-453, 1976.

[THM 83] *From denotational to operational and axiomatic semantics for ALGOL-like languages: An overview.* B. A. Trakhtenbrot, J. Y. Halpern, A. R. Meyer. Logics of Programs. Proceedings. Lecture Notes in Computer Science 164, 474-500, 1983.

[Urq 86] *Many-valued logic.* A. Urquhart. Handbook of Philosophical Logic, Vol. III, 71-116. D. Reidel Publishing Company, 1986.