# Elementos de lógica formalizados en Isabelle/HOL

Escuela Técnica Superior de Ingeniería Informática
Departamento de Ciencias de la Computación e Inteligencia Artificial
Trabajo Fin de Máster

**Sofía Santiago Fernández**

El presente Trabajo Fin de Grado se ha realizado en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Sevilla y ha sido supervisado por José Antonio Alonso Jiménez y María José Hidalgo Doblado.

# Índice

# Sumario


El objetivo de la Lógica es la formalización del conocimiento y su razonamiento. En este trabajo, estudiaremos elementos de la lógica proposicional desde la perspectiva teórica de *First−Order Logic and Automated Theorem Proving* [4] de Melvin Fitting. En particular, nos centraremos en la sintaxis y la semántica, concluyendo con la versión proposicional del lema de Hintikka sobre la satisfacibilidad de una clase determinada de conjuntos de fórmulas. Siguiendo la inspiración de *Propositional Proof Systems* [10] por Julius Michaelis y Tobias Nipkow, los resultados expuestos serán formalizados mediante Isabelle: un demostrador interactivo que incluye herramientas de razonamiento automático para guiar al usuario en el proceso de formalización, verificación y automatización de resultados. Concretamente, Isabelle/HOL es una especialización de Isabelle para la lógica de orden superior. Las demostraciones de los resultados en Isabelle/HOL se elaborarán siguiendo dos tácticas distintas a lo largo del trabajo. En primer lugar, cada lema será probado de manera detallada prescindiendo de toda herramienta de razonamiento automático, como resultado de una búsqueda inversa en cada paso de la prueba. En contraposición, elaboraremos una demostración automática alternativa de cada resultado que utilice todas las herramientas de razonamiento automático que proporciona el demostrador. De este modo, se evidenciará la capacidad de razonamiento automático de Isabelle.

Logic's purpose is about knowledge's formalisation and its reasoning. In this project, we will approach Propositional Logic's elements from the theoretical perspective of *First−Order Logic and Automated Theorem Proving* [4] by Melvin Fitting. We will focus on the study of Syntax and Semantics to reach propositional version of Hintikka's lemma, which determinate the satisfiability of a concrete type of formula set. Inspired by *Propositional Proof Systems* [10] by Julius Michaelis and Tobias Nipkow, these results will be formalised using Isabelle: a proof assistant including automatic reasoning tools to guide the user on formalising, verifying and automating results. In particular, Isabelle/HOL is the specialization of Isabelle for High-Order Logic. The processing of the results formalised in Isabelle/HOL follows two directions. In the first place, each lemma will be proved on detail without any automation, as the result of an inverse research on every step of the demonstration until it is only completed with deductions based on elemen-

tary rules and definitions. Conversely, we will alternatively prove the results using all the automatic reasoning tools that are provide by the proof assistant. In this way, Isabelle's power of automatic reasoning will be shown as the contrast between these two different proving tactics.

# Introducción

El objetivo de la Lógica es la formalización del conocimiento y el razonamiento sobre el mismo. Tiene su origen en la Antigua Grecia con Aristóteles y su investigación acerca de los principios del razonamiento válido o correcto, recogidos fundamentalmente en su obra *Organon*. De este modo, dio lugar a la lógica silogística, que consistía en la deducción de conclusiones a partir de dos premisas iniciales.

Posteriormente, los estoicos (400-200 a.C) comenzaron a cuestionarse temas relacionados con la semántica, como la naturaleza de la verdad. Formularon la *paradoja del mentiroso*, que plantea una incongruencia acerca de la veracidad del siguiente predicado.

*Esta oración es falsa*.

Sin embargo, no fue hasta el siglo XVII que el matemático y filósofo Gottfried Wilhelm Leibniz (1646 – 1716) instaura un programa lógico que propone la búsqueda de un sistema simbólico del lenguaje natural junto con la matematización del concepto de validez. Estas ideas fueron la principal motivación del desarrollo de la lógica moderna del siglo XIX de la mano de matemáticos y filósofos como Bernard Bolzano (1781 – 1848), George Boole (1815 – 1864), Charles Saunders Pierce (1839 – 1914) y Gottlob Frege (1848 – 1925). Fue este último quien introdujo el primer tratamiento sistemático de la lógica proposicional. Frege basó su tesis en el desarrollo de una sintaxis completa que combina el razonamiento de deducción de la silogística aristotélica con la noción estoica de conectivas para relacionar ideas. Paralelamente desarrolló una semántica asociada a dicha sintaxis que permitiese verificar la validez de los procesos deductivos. La lógica proposicional de Frege formó parte de la escuela denominada logicismo. Su objetivo consistía en investigar los fundamentos de las matemáticas con el fin de formalizarlos lógicamente, para así realizar deducciones y razonamientos válidos.

En las últimas décadas, el desarrollo de la computación y la inteligencia artificial ha permitido la formalización de las matemáticas y la lógica mediante el lenguaje computacional. Concretamente, el razonamiento automático es un área que investiga los distintos aspectos del razonamiento con el fin de crear programas y algoritmos para razonar de manera prácticamente automática. Se fundamenta en el programa lógico desarrollado por Leibniz, estructurado en base a dos principios: la formalización rigurosa de resultados y el desarrollo de algoritmos que permitan manipular y razonar a partir de dichas formalizaciones. Entre las principales aplicaciones de este áres se encuentra la verificación y síntesis automáticas de programas. De este modo, podemos validar distintos razonamientos, así como crear herramientas de razonamiento automático que permitan el desarrollo de nuevos resultados.

En este contexto nace Isabelle en 1986, desarrollada por Larry Paulson de la Universidad de Cambridge y Tobias Nipkow del Techniche Universität München. Isabelle es un demostrador interactivo que, desde el razonamiento automático, facilita la formalización lógica de resultados y proporciona herramientas para realizar deducciones. En particular, Isabelle/HOL es la especialización de Isabelle para la lógica de orden superior. Junto con Coq, ACL2 y PVS, entre otros, constituye uno de los demostradores interactivos más influyentes.

Como demostrador interactivo, Isabelle permite automatizar razonamientos guiados por el usuario, verificando cada paso de una deducción de manera precisa. Además, incorpora herramientas de razonamiento automático para mejorar la productividad del proceso de demostración. Para ello, cuenta con una extensa librería de resultados lógicos y matemáticos que han sido formalizados y continúan en desarrollo por parte de proyectos como *The Alexandria Project*: *Large−Scale Formal Proof for the Working Mathematician*. Este proyecto comienza en 2017, dirigido por Lawrence Paulson desde la Universidad de Cambridge. Tiene como finalidad la formalización de distintas teorías para ampliar la librería de Isabelle, junto con la creación de herramientas interactivas que asistan a los matemáticos en el proceso de formalización, demostración y búsqueda de nuevos resultados.

El objetivo de este trabajo es la formalización de elementos y resultados destacados de la lógica proposicional en Isabelle/HOL. Está inspirado en la primera sección de la publicación *Propositional Proof Systems* [*10*] de Julius Michaelis y Tobias Nipkow. Del mismo modo, cabe citar los artículos *Constructive Formalization of Classical Modal Logic* [*3*] de Christian Doczkal y Gert Smolka, y *Propositional Calculus in Coq* [*13*] de Floris van Doorn, por la influencia ejercida en el desarrollo de este trabajo. El contenido teórico del mismo se fundamenta en el libro *First−Order Logic and Automated Theorem Proving* [*4*] de Melvin Fitting. Los tres capítulos tratados consisten en la sintaxis, semántica y, finalmente, la versión proposicional del lema de Hintikka. Este último fue desarrollado por el filósofo y lógico Jaakko Hintikka (1929- 2015) como herramienta para probar la completitud de la lógica de primer orden.

En el primer capítulo sobre sintaxis se establecen inicialmente las variables proposicionales que conforman los elementos básicos del alfabeto, junto con una serie de conectivas que actúan sobre ellas. De este modo, se define por recursión el conjunto de las fórmulas proposicionales como el menor conjunto de estructuras sintácticas con dicho alfabeto y conectivas que contiene a las fórmulas básicas (una constante $\bot$ y las propias variables proposicionales, llamadas fórmulas atómicas) y es cerrado mediante procedimientos de formación de nuevas fórmulas a partir de otras, en los que intervienen las conectivas. Como es habitual, dada esta definición recursiva, se dispone de un esquema de inducción sobre fórmulas que nos permitirá probar los resultados expuestos. Del mismo modo, se define recursivamente el conjunto de subfórmulas de una fórmula, mostrando propiedades que describen la estructura de las mismas en relación con las propias fórmulas. Finalmente se presenta la fórmula $\top$ a partir de la constante $\bot$, y dos conectivas generalizadas que permiten extender conectivas binarias a una lista de fórmulas.

En el siguiente capítulo precisamos la semántica asociada a las estructuras sintácticas. Para ello, se define una interpretación como una aplicación que asocia un booleano a cada variable proposicional. Por recursión sobre la estructura de las fórmulas proposicionales, podemos definir el valor de una fórmula en una interpretación dada. De este modo, se prueba que dicho valor queda unívocamente determinado por la imagen que la interpretación asocia a cada variable proposicional que aparece en la fórmula. Las nociones semánticas se extienden análogamente a las fórmulas formadas con conectivas generalizadas.

Posteriormente se introducen dos definiciones semánticas fundamentales: modelo de una fórmula y fórmula satisfacible. La primera hace referencia a una interpretación en la que el valor de una fórmula dada es verdadero, mientras la segunda se trata de una fórmula para la que existe una interpretación que sea modelo suyo. Ambas nociones se extienden a conjuntos de fórmulas. Por otro lado, se define el concepto de tautología como aquella fórmula cuyo valor es verdadero en toda interpretación. Para concluir la sección, daremos una noción formal de consecuencia lógica.

El capítulo tercero, y último, tiene como objetivo probar el lema de Hintikka, que manifiesta la satisfacibilidad de ciertos conjuntos de fórmulas. Para ello define dicha clase de conjuntos, llamados conjuntos de Hintikka, de modo que para cada uno de ellos se determina paralelamente una interpretación asociada, garantizando que esta es modelo de cada fórmula del conjunto.

En lo referente a las demostraciones asistidas por Isabelle/HOL de los resultados formalizados a lo largo de las secciones, se elaborarán dos tipos de pruebas correspondientes a dos tácticas distintas. En primer lugar, se probará cada resultado siguiendo un esquema de demostración detallado. En él utilizaremos únicamente y de manera precisa las reglas de simplificación y definiciones incluidas en la librería de Isabelle, prescindiendo de las herramientas de razonamiento automático del demostrador. Para ello, se realiza una búsqueda inversa en cada paso de la demostración automática hasta llegar a un desarrollo de la prueba basado en deducciones a partir de resultados elementales que la completen de manera rigurosa. En contraposición, se evidenciará la capacidad de razonamiento automático de Isabelle/HOL mediante la realización de una prueba alternativa siguiendo un esquema de demostración automático. Para ello se utilizarán las herramientas de razonamiento que han sido elaboradas en Isabelle/HOL con el objetivo de realizar deducciones de la manera más eficiente.

*Este trabajo está disponible en la plataforma* GitHub *mediante el siguiente enlace*:

https://github.com/sofsanfer/TFG

# Capítulo 1

# Notación uniforme

**Comentario 1**: Localización de sello.png.

**Comentario 2**: Cambiar los directores

**Comentario 3**: Introducción. Mirar fitting p. 53 y 54

En esta sección introduciremos la notación uniforme inicialmente desarrollada por *R. M. Smullyan* (añadir referencia bibliográfica). La finalidad de dicha notación es reducir el número de casos a considerar sobre la estructura de las fórmulas al clasificar éstas en dos categorías, facilitando las demostraciones y métodos empleados en adelante.

**Comentario 4**: Añadir referencia bibliográfica.

De este modo, las fórmulas proposicionales pueden ser de dos tipos: aquellas que de tipo conjuntivo (las fórmulas $\alpha$) y las de tipo disyuntivo (las fórmulas $\beta$). Cada fórmula de tipo $\alpha$, o $\beta$ respectivamente, tiene asociada sus dos componentes $\alpha_1$ y $\alpha_2$, o $\beta_1$ y $\beta_2$ respectivamente. Para justificar dicha clasificación, introduzcamos inicialmente la definición de fórmulas semánticamente equivalentes.

**Definición 1.0.1** *Dos fórmulas son semánticamente equivalentes si tienen el mismo valor para toda interpretación.*

En Isabelle podemos formalizar la definición de la siguiente manera.

**definition** *semanticEq F G* $\equiv \forall\, \mathcal{A}.\ (\mathcal{A} \models F) \longleftrightarrow (\mathcal{A} \models G)$

De este modo, según la definición del valor de verdad de una fórmula proposicional en una interpretación dada, podemos ver los siguientes ejemplos de fórmulas semánticamente equivalentes.

**lemma** *semanticEq* $(Atom\ p)\ ((Atom\ p) \vee (Atom\ p))$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $(Atom\ p)\ ((Atom\ p) \wedge (Atom\ p))$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $\bot\ (\bot \wedge \bot)$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $\bot\ (\bot \vee \bot)$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $\bot\ (\neg\ \top)$
**by** $($*simp add*: *semanticEq-def top-semantics*$)$

**lemma** *semanticEq* $F\ (\neg(\neg\ F))$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $(\neg(\neg\ F))\ (F \vee F)$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $(\neg(\neg\ F))\ (F \wedge F)$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $(\neg\ F \wedge \neg\ G)\ (\neg(F \vee G))$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** *semanticEq* $(F \longrightarrow G)\ (\neg\ F \vee G)$
**by** $($*simp add*: *semanticEq-def*$)$

En contraposición, también podemos dar ejemplos de fórmulas que no son semánticamente equivalentes.

**lemma** $\neg$ *semanticEq* $(Atom\ p)\ (\neg(Atom\ p))$
**by** $($*simp add*: *semanticEq-def*$)$

**lemma** $\neg$ *semanticEq* $\bot\ \top$
**by** $($*simp add*: *semanticEq-def top-semantics*$)$

Por tanto, diremos intuitivamente que una fórmula es de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ si es semánticamente equivalente a la fórmula $\alpha_1 \wedge \alpha_2$. Del mismo modo, una fórmula será de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ si es semánticamente equivalente a la fórmula $\beta_1 \vee \beta_2$.

**Definición 1.0.2** *Las fórmulas de tipo $\alpha$ (fórmulas conjuntivas) y sus correspondientes componentes $\alpha_1$ y $\alpha_2$ se definen como sigue: dadas F y G fórmulas cualesquiera,*

1. *$F \wedge G$ es una fórmula de tipo $\alpha$ cuyas componentes son F y G.*
2. *$\neg(F \vee G)$ es una fórmula de tipo $\alpha$ cuyas componentes son $\neg F$ y $\neg G$.*
3. *$\neg(F \longrightarrow G)$ es una fórmula de tipo $\alpha$ cuyas componentes son F y $\neg G$.*

De este modo, de los ejemplos anteriores podemos deducir que las fórmulas atómicas son de tipo $\alpha$ y sus componentes $\alpha_1$ y $\alpha_2$ son la propia fórmula. Del mismo modo, la constante $\perp$ también es una fórmula conjuntiva cuyas componentes son ella misma. Por último, podemos observar que dada una fórmula cualquiera $F$, su doble negación $\neg(\neg F)$ es una fórmula de tipo $\alpha$ y componentes $F$ y $F$.

Formalizaremos en Isabelle el conjunto de fórmulas $\alpha$ como un predicato inductivo. De este modo, las reglas anteriores que construyen el conjunto de fórmulas de tipo $\alpha$ se formalizan en Isabelle como reglas de introducción. Además, añadiremos explícitamente una cuarta regla que introduce la doble negación de una fórmula como fórmula de tipo $\alpha$. De este modo, facilitaremos la prueba de resultados posteriores relacionados con la definición de conjunto de Hintikka, que constituyen una base para la demostración del *teorema de existencia de modelo*.

**inductive** *Con* :: *'a formula => 'a formula => 'a formula => bool* **where**
*Con* (*And F G*) *F G* |
*Con* (*Not* (*Or F G*)) (*Not F*) (*Not G*) |
*Con* (*Not* (*Imp F G*)) *F* (*Not G*) |
*Con* (*Not* (*Not F*)) *F F*

Las reglas de introducción que proporciona la definición anterior son las siguientes.

$$\begin{aligned}
&Con\ (F \wedge G)\ F\ G \\
&Con\ (\neg\ (F \vee G))\ (\neg\ F)\ (\neg\ G) \\
&Con\ (\neg\ (F \to G))\ F\ (\neg\ G) \\
&Con\ (\neg\ (\neg\ F))\ F\ F
\end{aligned} \qquad (Con.intros)$$

Por otro lado, definamos las fórmulas disyuntivas.

**Definición 1.0.3** *Las fórmulas de tipo $\beta$ (fórmulas disyuntivas) y sus correspondientes componentes $\beta_1$ y $\beta_2$ se definen como sigue: dadas F y G fórmulas cualesquiera,*

1. *$F \vee G$ es una fórmula de tipo $\beta$ cuyas componentes son $F$ y $G$.*

2. *$F \longrightarrow G$ es una fórmula de tipo $\beta$ cuyas componentes son $\neg F$ y $G$.*

3. *$\neg(F \wedge G)$ es una fórmula de tipo $\beta$ cuyas componentes son $\neg F$ y $\neg G$.*

De los ejemplos dados anteriormente, podemos deducir análogamente que las fórmulas atómicas, la constante $\bot$ y la doble negación sob también fórmulas disyuntivas con las mismas componentes que las dadas para el tipo conjuntivo.

Del mismo modo, su formalización se realiza como un predicado inductivo, de manera que las reglas que definen el conjunto de fórmulas de tipo $\beta$ se formalizan en Isabelle como reglas de introducción. Análogamente, introduciremos de manera explícita una regla que señala que la doble negación de una fórmula es una fórmula de tipo disyuntivo.

**inductive** *Dis* :: *'a formula => 'a formula => 'a formula => bool* **where**
*Dis (Or F G) F G |*
*Dis (Imp F G) (Not F) G |*
*Dis (Not (And F G)) (Not F) (Not G) |*
*Dis (Not (Not F)) F F*

Del mismo modo, las reglas de introducción que proporciona esta formalización se muestran a continuación.

$$
\begin{aligned}
&Dis\ (F \vee G)\ F\ G \\
&Dis\ (F \rightarrow G)\ (\neg\ F)\ G \\
&Dis\ (\neg\ (F \wedge G))\ (\neg\ F)\ (\neg\ G) \\
&Dis\ (\neg\ (\neg\ F))\ F\ F
\end{aligned}
\qquad (Dis.intros)
$$

Cabe observar que las formalizaciones de la definiciones de fórmulas de tipo $\alpha$ y $\beta$ son definiciones sintácticas, pues construyen los correspondientes conjuntos de fórmulas a partir de una reglas sintácticas concretas. Se trata de una simplificación de la intuición original de la clasificación de las fórmulas mediante notación uniforme, ya que se prescinde de la noción de equivalencia semántica que permite clasificar la totalidad de las fórmulas proposicionales.

Veamos la clasificación de casos concretos de fórmulas. Por ejemplo, según hemos definido la fórmula $\top$, es sencillo comprobar que se trata de una fórmula disyuntiva.

**lemma** *Dis* $\top$ $(\neg \bot)$ $\bot$
**unfolding** *Top-def* **by** (*simp only*: *Dis.intros*(2))

Por otro lado, se observa a partir de las correspondientes definiciones que la conjunción generalizada de una lista de fórmulas es una fórmula de tipo $\alpha$ y la disyunción generalizada de una lista de fórmulas es una fórmula de tipo $\beta$.

**lemma** *Con* $(\bigwedge(F\#Fs))$ *F* $(\bigwedge Fs)$
**by** (*simp only*: *BigAnd.simps Con.intros*(*1*))

**lemma** *Dis* $(\bigvee(F\#Fs))$ *F* $(\bigvee Fs)$
**by** (*simp only*: *BigOr.simps Dis.intros*(*1*))

Finalmente, de las reglas que definen las fórmulas conjuntivas y disyuntivas se deduce que la doble negación de una fórmula es una fórmula perteneciente a ambos tipos.

**lemma** *notDisCon*: *Con* (*Not* (*Not F*)) *F F Dis* (*Not* (*Not F*)) *F F*
**by** (*simp only*: *Con.intros*(*4*) *Dis.intros*(*4*))+

A continuación vamos a introducir el siguiente lema que caracteriza las fórmulas de tipo $\alpha$ y $\beta$, facilitando el uso de la notación uniforme en Isabelle.

**lemma** *con-dis-simps*:
$Con\ a1\ a2\ a3 = (a1 = a2 \wedge a3 \vee$
$(\exists F\ G.\ a1 = \neg\ (F \vee G) \wedge a2 = \neg\ F \wedge a3 = \neg\ G) \vee$
$(\exists G.\ a1 = \neg\ (a2 \longrightarrow G) \wedge a3 = \neg\ G) \vee$
$a1 = \neg\ (\neg\ a2) \wedge a3 = a2)$
$Dis\ a1\ a2\ a3 = (a1 = a2 \vee a3 \vee$
$(\exists F\ G.\ a1 = F \longrightarrow G \wedge a2 = \neg\ F \wedge a3 = G) \vee$
$(\exists F\ G.\ a1 = \neg\ (F \wedge G) \wedge a2 = \neg\ F \wedge a3 = \neg\ G) \vee$
$a1 = \neg\ (\neg\ a2) \wedge a3 = a2)$
**by** (*simp-all add*: *Con.simps Dis.simps*)

Por último, introduzcamos resultados que permiten caracterizar los conjuntos de Hintikka y la propiedad de consistencia proposicional empleando la notación uniforme.

**Lema 1.0.4 (Caracterización de los conjuntos de Hintikka mediante la notación uniforme)** *Dado un conjunto de fórmulas proposicionales S, son equivalentes:*

1. *S es un conjunto de Hintikka.*
2. *Se verifican las condiciones siguientes:*
   - $\bot$ *no pertenece a S.*
   - *Dada p una fórmula atómica cualquiera, no se tiene simultáneamente que* $p \in S$ *y* $\neg\, p \in S$.

- *Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ se verifica que si la fórmula pertenece a $S$, entonces $\alpha_1$ y $\alpha_2$ también.*
- *Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ se verifica que si la fórmula pertenece a $S$, entonces o bien $\beta_1$ pertenece a $S$ o bien $\beta_2$ pertenece a $S$.*

En Isabelle/HOL se formaliza del siguiente modo.

**lemma** $\mathit{Hintikka}\ S = (\bot \notin S$
$\wedge\ (\forall k.\ \mathit{Atom}\ k \in S \longrightarrow \neg\ (\mathit{Atom}\ k) \in S \longrightarrow \mathit{False})$
$\wedge\ (\forall F\ G\ H.\ \mathit{Con}\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall F\ G\ H.\ \mathit{Dis}\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \vee H \in S))$
**oops**

Procedamos a la demostración del resultado.

**Demostración:** Para probar la equivalencia, veamos cada una de las implicaciones por separado.

*1)* $\Longrightarrow$ *2)*

Supongamos que $S$ es un conjunto de Hintikka. Vamos a probar que, en efecto, se verifican las condiciones del enunciado del lema.

Por definición de conjunto de Hintikka, $S$ verifica las siguientes condiciones:

1. $\bot \notin S$.
2. Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg\, p \in S$.
3. Si $G \wedge H \in S$, entonces $G \in S$ y $H \in S$.
4. Si $G \vee H \in S$, entonces $G \in S$ o $H \in S$.
5. Si $G \rightarrow H \in S$, entonces $\neg\, G \in S$ o $H \in S$.
6. Si $\neg(\neg\, G) \in S$, entonces $G \in S$.
7. Si $\neg(G \wedge H) \in S$, entonces $\neg\, G \in S$ o $\neg\, H \in S$.
8. Si $\neg(G \vee H) \in S$, entonces $\neg\, G \in S$ y $\neg\, H \in S$.
9. Si $\neg(G \rightarrow H) \in S$, entonces $G \in S$ y $\neg\, H \in S$.

De este modo, el conjunto $S$ cumple la primera y la segunda condición del enunciado del lema, que se corresponden con las dos primeras condiciones de la definición de conjunto de Hintikka. Veamos que, además, verifica las dos últimas condiciones del resultado.

En primer lugar, probemos que para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ se verifica que si la fórmula pertenece al conjunto $S$, entonces $\alpha_1$ y $\alpha_2$ también. Para ello, supongamos que una fórmula cualquiera de tipo $\alpha$ pertence a $S$. Por definición de este tipo de fórmulas, tenemos que $\alpha$ puede ser de la forma $G \wedge H$, $\neg(\neg G)$, $\neg(G \vee H)$ o $\neg(G \longrightarrow H)$ para fórmulas $G$ y $H$ cualesquiera. Probemos que, para cada tipo de fórmula $\alpha$ perteneciente a $S$, sus componentes $\alpha_1$ y $\alpha_2$ están en $S$.

- *Fórmula del tipo $G \wedge H$*: Sus componentes conjuntivas son $G$ y $H$. Por la tercera condición de la definición de conjunto de Hintikka, obtenemos que si $G \wedge H$ pertenece a $S$, entonces $G$ y $H$ están ambas en el conjunto, lo que prueba este caso.

- *Fórmula del tipo $\neg(\neg G)$*: Sus componentes conjuntivas son ambas $G$. Por la sexta condición de la definición de conjunto de Hintikka, obtenemos que si $\neg(\neg G)$ pertenece a $S$, entonces $G$ pertenece al conjunto, lo que prueba este caso.

- *Fórmula del tipo $\neg(G \vee H)$*: Sus componentes conjuntivas son $\neg G$ y $\neg H$. Por la octava condición de la definición de conjunto de Hintikka, obtenemos que si $\neg(G \vee H)$ pertenece a $S$, entonces $\neg G$ y $\neg H$ están ambas en el conjunto, lo que prueba este caso.

- *Fórmula del tipo $\neg(G \longrightarrow H)$*: Sus componentes conjuntivas son $G$ y $\neg H$. Por la novena condición de la definición de conjunto de Hintikka, obtenemos que si $\neg(G \longrightarrow H)$ pertenece a $S$, entonces $G$ y $\neg H$ están ambas en el conjunto, lo que prueba este caso.

Finalmente, probemos que para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ se verifica que si la fórmula pertenece al conjunto $S$, entonces o bien $\beta_1$ pertenece al conjunto o bien $\beta_2$ pertenece a conjunto. Para ello, supongamos que una fórmula cualquiera de tipo $\beta$ pertence a $S$. Por definición de este tipo de fórmulas, tenemos que $\beta$ puede ser de la forma $G \vee H$, $G \longrightarrow H$, $\neg(\neg G)$ o $\neg(G \wedge H)$ para fórmulas $G$ y $H$ cualesquiera. Probemos que, para cada tipo de fórmula $\beta$ perteneciente a $S$, o bien su componente $\beta_1$ pertenece a $S$ o bien su componente $\beta_2$ pertenece a $S$.

- *Fórmula del tipo $G \vee H$*: Sus componentes disyuntivas son $G$ y $H$. Por la cuarta condición de la definición de conjunto de Hintikka, obtenemos que si $G \vee H$ pertenece a $S$, entonces o bien $G$ está en $S$ o bien $H$ está en $S$, lo que prueba este caso.

- *Fórmula del tipo $G \longrightarrow H$*: Sus componentes disyuntivas son $\neg G$ y $H$. Por la quinta condición de la definición de conjunto de Hintikka, obtenemos que si $G \longrightarrow H$ pertenece a $S$, entonces o bien $\neg G$ pertenece al conjunto o bien $H$ pertenece al conjunto, lo que prueba este caso.

- *Fórmula del tipo $\neg(\neg G)$*: Sus componentes conjuntivas son ambas $G$. Por la sexta condición de la definición de conjunto de Hintikka, obtenemos que si $\neg(\neg G)$ pertenece a $S$, entonces $G$ pertenece al conjunto. De este modo, por la regla de introducción a la disyunción, se prueba que o bien una de las componentes está en el conjunto o bien lo está la otra pues, en este caso, coinciden.

▪ *Fórmula del tipo* $\neg(G \wedge H)$: Sus componentes conjuntivas son $\neg G$ y $\neg H$. Por la séptima condición de la definición de conjunto de Hintikka, obtenemos que si $\neg(G \wedge H)$ pertenece a $S$, entonces o bien $\neg G$ pertenece al conjunto o bien $\neg H$ pertenece al conjunto, lo que prueba este caso.

*2)* $\Longrightarrow$ *1)*

Supongamos que se verifican las condiciones del enunciado del lema:

- $\perp$ no pertenece a $S$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ se verifica que si la fórmula pertenece a $S$, entonces $\alpha_1$ y $\alpha_2$ también.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ se verifica que si la fórmula pertenece a $S$, entonces o bien $\beta_1$ pertenece a $S$ o bien $\beta_2$ pertenece a $S$.

Vamos a probar que $S$ es un conjunto de Hintikka.

Por la definición de conjunto de Hintikka, es suficiente probar las siguientes condiciones:

1. $\perp \notin S$.
2. Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$.
3. Si $G \wedge H \in S$, entonces $G \in S$ y $H \in S$.
4. Si $G \vee H \in S$, entonces $G \in S$ o $H \in S$.
5. Si $G \rightarrow H \in S$, entonces $\neg G \in S$ o $H \in S$.
6. Si $\neg(\neg G) \in S$, entonces $G \in S$.
7. Si $\neg(G \wedge H) \in S$, entonces $\neg G \in S$ o $\neg H \in S$.
8. Si $\neg(G \vee H) \in S$, entonces $\neg G \in S$ y $\neg H \in S$.
9. Si $\neg(G \rightarrow H) \in S$, entonces $G \in S$ y $\neg H \in S$.

En primer lugar se observa que, por hipótesis, se verifican las dos primeras condiciones de la definición de conjunto de Hintikka. Veamos que, en efecto, se cumplen las demás.

3) Supongamos que $G \wedge H$ está en $S$ para fórmulas $G$ y $H$ cualesquiera. Por definición, $G \wedge H$ es una fórmula de tipo $\alpha$ con componentes $G$ y $H$. Por lo tanto, por hipótesis se cumple que $G$ y $H$ están en $S$.

4) Supongamos que $G \vee H$ está en $S$ para fórmulas $G$ y $H$ cualesquiera. Por definición, $G \vee H$ es una fórmula de tipo $\beta$ con componentes $G$ y $H$. Por lo tanto, por hipótesis se cumple que o bien $G$ está en $S$ o bien $H$ está en $S$.

5) Supongamos que $G \longrightarrow H$ está en $S$ para fórmulas $G$ y $H$ cualesquiera. Por definición, $G \longrightarrow H$ es una fórmula de tipo $\beta$ con componentes $\neg\, G$ y $H$. Por lo tanto, por hipótesis se cumple que o bien $\neg\, G$ está en $S$ o bien $H$ está en $S$.

6) Supongamos que $\neg(\neg\, G)$ está en $S$ para una fórmula $G$ cualquiera. Por definición, $\neg(\neg\, G)$ es una fórmula de tipo $\alpha$ cuyas componentes son ambas $G$. Por lo tanto, por hipótesis se cumple que $G$ está en $S$.

7) Supongamos que $\neg(G \wedge H)$ está en $S$ para fórmulas $G$ y $H$ cualesquiera. Por definición, $\neg(G \wedge H)$ es una fórmula de tipo $\beta$ con componentes $\neg\, G$ y $\neg\, H$. Por lo tanto, por hipótesis se cumple que o bien $\neg\, G$ está en $S$ o bien $\neg\, H$ está en $S$.

8) Supongamos que $\neg(G \vee H)$ está en $S$ para fórmulas $G$ y $H$ cualesquiera. Por definición, $\neg(G \vee H)$ es una fórmula de tipo $\alpha$ con componentes $\neg\, G$ y $\neg\, H$. Por lo tanto, por hipótesis se cumple que $\neg\, G$ y $\neg\, H$ están en $S$.

9) Supongamos que $\neg(G \longrightarrow H)$ está en $S$ para fórmulas $G$ y $H$ cualesquiera. Por definición, $\neg(G \longrightarrow H)$ es una fórmula de tipo $\alpha$ con componentes $G$ y $\neg\, H$. Por lo tanto, por hipótesis se cumple que $G$ y $\neg\, H$ están en $S$.

Por tanto, queda probado el resultado.

□

Para probar de manera detallada el lema en Isabelle vamos a demostrar cada una de las implicaciones de la equivalencia por separado.

La primera implicación del lema se basa en dos lemas auxiliares. El primero de ellos prueba que la tercera, sexta, octava y novena condición de la definición de conjunto de Hintikka son suficientes para probar que para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ se verifica que si la fórmula pertenece al conjunto $S$, entonces $\alpha_1$ y $\alpha_2$ también. Su demostración detallada en Isabelle se muestra a continuación.

**lemma** *Hintikka-alt1Con*:

```
 assumes (∀G H. G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S)
 ∧ (∀G. ¬ (¬ G) ∈ S ⟶ G ∈ S)
 ∧ (∀G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S)
 ∧ (∀G H. ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S)
 shows Con F G H ⟶ F ∈ S ⟶ G ∈ S ∧ H ∈ S
proof (rule impI)
 assume Con F G H
 then have F = G ∧ H ∨
  ((∃G1 H1. F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1) ∨
  (∃H2. F = ¬ (G → H2) ∧ H = ¬ H2) ∨
  F = ¬ (¬ G) ∧ H = G)
  by (simp only: con-dis-simps(1))
 thus F ∈ S ⟶ G ∈ S ∧ H ∈ S
 proof (rule disjE)
  assume F = G ∧ H
  have ∀G H. G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S
   using assms by (rule conjunct1)
  thus F ∈ S ⟶ G ∈ S ∧ H ∈ S
   using ‹F = G ∧ H› by (iprover elim: allE)
 next
  assume (∃G1 H1. F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1) ∨
  ((∃H2. F = ¬ (G → H2) ∧ H = ¬ H2) ∨
  F = ¬ (¬ G) ∧ H = G)
  thus F ∈ S ⟶ G ∈ S ∧ H ∈ S
  proof (rule disjE)
   assume E1:∃G1 H1. F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1
   obtain G1 H1 where A1:F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1
    using E1 by (iprover elim: exE)
   then have F = ¬ (G1 ∨ H1)
    by (rule conjunct1)
   have G = ¬ G1
    using A1 by (iprover elim: conjunct1)
   have H = ¬ H1
    using A1 by (iprover elim: conjunct1)
   have ∀G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S
    using assms by (iprover elim: conjunct2 conjunct1)
   thus F ∈ S ⟶ G ∈ S ∧ H ∈ S
    using ‹F = ¬ (G1 ∨ H1)› ‹G = ¬ G1› ‹H = ¬ H1› by (iprover elim: allE)
  next
   assume (∃H2. F = ¬ (G → H2) ∧ H = ¬ H2) ∨
```

$F = \neg\ (\neg\ G) \wedge H = G$
**thus** $F \in S \longrightarrow G \in S \wedge H \in S$
**proof** (*rule disjE*)
**assume** $E2{:}\exists\, H2.\ F = \neg\ (G \to H2) \wedge H = \neg\ H2$
**obtain** $H2$ **where** $A2{:}F = \neg\ (G \to H2) \wedge H = \neg\ H2$
**using** $E2$ **by** (*rule exE*)
**have** $F = \neg\ (G \to H2)$
**using** $A2$ **by** (*rule conjunct1*)
**have** $H = \neg\ H2$
**using** $A2$ **by** (*rule conjunct2*)
**have** $\forall\, G\, H.\ \neg(G \to H) \in S \longrightarrow G \in S \wedge \neg\ H \in S$
**using** *assms* **by** (*iprover elim*: *conjunct2 conjunct1*)
**thus** $F \in S \longrightarrow G \in S \wedge H \in S$
**using** $\langle F = \neg\ (G \to H2)\rangle$ $\langle H = \neg\ H2\rangle$ **by** (*iprover elim*: *allE*)
**next**
**assume** $F = \neg\ (\neg\ G) \wedge H = G$
**then have** $F = \neg\ (\neg\ G)$
**by** (*rule conjunct1*)
**have** $H = G$
**using** $\langle F = \neg\ (\neg\ G) \wedge H = G\rangle$ **by** (*rule conjunct2*)
**have** $\forall\, G.\ \neg\ (\neg\ G) \in S \longrightarrow G \in S$
**using** *assms* **by** (*iprover elim*: *conjunct2 conjunct1*)
**then have** $\neg\ (\neg\ G) \in S \longrightarrow G \in S$
**by** (*rule allE*)
**then have** $F \in S \longrightarrow G \in S$
**by** (*simp only*: $\langle F = \neg\ (\neg\ G)\rangle$)
**then have** $F \in S \longrightarrow G \in S \wedge G \in S$
**by** (*simp only*: *conj-absorb*)
**thus** $F \in S \longrightarrow G \in S \wedge H \in S$
**by** (*simp only*: $\langle H{=}G\rangle$)
**qed**
**qed**
**qed**
**qed**

Por otro lado, el segundo lema auxiliar prueba que la cuarta, quinta, sexta y séptima condición de la definición de conjunto de Hintikka son suficientes para probar que para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ se verifica que si la fórmula pertenece al conjunto $S$, entonces o bien $\beta_1$ pertenece al conjunto o bien $\beta_2$ pertenece al conjunto. Veamos su prueba detallada en Isabelle/HOL.

**lemma** *Hintikka-alt1Dis*:

**assumes** $(\forall\ G\,H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S)$
$\wedge\ (\forall\ G\,H.\ G \rightarrow H \in S \longrightarrow \neg\, G \in S \vee H \in S)$
$\wedge\ (\forall\ G.\ \neg\,(\neg\, G) \in S \longrightarrow G \in S)$
$\wedge\ (\forall\ G\,H.\ \neg(G \wedge H) \in S \longrightarrow \neg\, G \in S \vee \neg\, H \in S)$
**shows** $\mathit{Dis}\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \vee H \in S$
**proof** (*rule impI*)
**assume** $\mathit{Dis}\ F\ G\ H$
**then have** $F = G \vee H\ \vee$
$(\exists\, G1\ H1.\ F = G1 \rightarrow H1 \wedge G = \neg\, G1 \wedge H = H1)\ \vee$
$(\exists\, G2\ H2.\ F = \neg\,(G2 \wedge H2) \wedge G = \neg\, G2 \wedge H = \neg\, H2)\ \vee$
$F = \neg\,(\neg\, G) \wedge H = G$
**by** (*simp only*: *con-dis-simps*(2))
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**proof** (*rule disjE*)
**assume** $F = G \vee H$
**have** $\forall\, G\,H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S$
**using** *assms* **by** (*rule conjunct1*)
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**using** $\langle F = G \vee H\rangle$ **by** (*iprover elim*: *allE*)
**next**
**assume** $(\exists\, G1\ H1.\ F = G1 \rightarrow H1 \wedge G = \neg\, G1 \wedge H = H1)\ \vee$
$(\exists\, G2\ H2.\ F = \neg\,(G2 \wedge H2) \wedge G = \neg\, G2 \wedge H = \neg\, H2)\ \vee$
$F = \neg\,(\neg\, G) \wedge H = G$
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**proof** (*rule disjE*)
**assume** $E1{:}\exists\, G1\ H1.\ F = G1 \rightarrow H1 \wedge G = \neg\, G1 \wedge H = H1$
**obtain** $G1\ H1$ **where** $A1{:}F = G1 \rightarrow H1 \wedge G = \neg\, G1 \wedge H = H1$
**using** $E1$ **by** (*iprover elim*: *exE*)
**have** $F = G1 \rightarrow H1$
**using** $A1$ **by** (*rule conjunct1*)
**have** $G = \neg\, G1$
**using** $A1$ **by** (*iprover elim*: *conjunct1*)
**have** $H = H1$
**using** $A1$ **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** $\forall\, G\,H.\ G \rightarrow H \in S \longrightarrow \neg\, G \in S \vee H \in S$
**using** *assms* **by** (*iprover elim*: *conjunct2 conjunct1*)
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**using** $\langle F = G1 \rightarrow H1\rangle$ $\langle G = \neg\, G1\rangle$ $\langle H = H1\rangle$ **by** (*iprover elim*: *allE*)
**next**
**assume** $(\exists\, G2\ H2.\ F = \neg\,(G2 \wedge H2) \wedge G = \neg\, G2 \wedge H = \neg\, H2)\ \vee$

$F = \neg (\neg G) \wedge H = G$
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**proof** (*rule disjE*)
**assume** $E2{:}\exists G2\ H2.\ F = \neg (G2 \wedge H2) \wedge G = \neg G2 \wedge H = \neg H2$
**obtain** $G2\ H2$ **where** $A2{:}F = \neg (G2 \wedge H2) \wedge G = \neg G2 \wedge H = \neg H2$
**using** $E2$ **by** (*iprover elim*: *exE*)
**have** $F = \neg (G2 \wedge H2)$
**using** $A2$ **by** (*rule conjunct1*)
**have** $G = \neg G2$
**using** $A2$ **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** $H = \neg H2$
**using** $A2$ **by** (*iprover elim*: *conjunct1*)
**have** $\forall\ G\ H.\ \neg(G \wedge H) \in S \longrightarrow \neg G \in S \vee \neg H \in S$
**using** *assms* **by** (*iprover elim*: *conjunct2 conjunct1*)
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**using** $\langle F = \neg(G2 \wedge H2)\rangle$ $\langle G = \neg G2\rangle$ $\langle H = \neg H2\rangle$ **by** (*iprover elim*: *allE*)
**next**
**assume** $F = \neg (\neg G) \wedge H = G$
**then have** $F = \neg (\neg G)$
**by** (*rule conjunct1*)
**have** $H = G$
**using** $\langle F = \neg (\neg G) \wedge H = G\rangle$ **by** (*rule conjunct2*)
**have** $\forall\ G.\ \neg (\neg G) \in S \longrightarrow G \in S$
**using** *assms* **by** (*iprover elim*: *conjunct2 conjunct1*)
**then have** $\neg (\neg G) \in S \longrightarrow G \in S$
**by** (*rule allE*)
**then have** $F \in S \longrightarrow G \in S$
**by** (*simp only*: $\langle F = \neg (\neg G)\rangle$)
**then have** $F \in S \longrightarrow G \in S \vee G \in S$
**by** (*simp only*: *disj-absorb*)
**thus** $F \in S \longrightarrow G \in S \vee H \in S$
**by** (*simp only*: $\langle H = G\rangle$)
**qed**
**qed**
**qed**
**qed**

Finalmente, podemos demostrar detalladamente esta primera implicación de la equivalencia del lema en Isabelle.

**lemma** *Hintikka-alt1*:
**assumes** *Hintikka S*

```
 shows ⊥ ∉ S
∧ (∀k. Atom k ∈ S ⟶ ¬ (Atom k) ∈ S ⟶ False)
∧ (∀F G H. Con F G H ⟶ F ∈ S ⟶ G ∈ S ∧ H ∈ S)
∧ (∀F G H. Dis F G H ⟶ F ∈ S ⟶ G ∈ S ∨ H ∈ S)
proof −
 have Hk:(⊥ ∉ S
 ∧ (∀k. Atom k ∈ S ⟶ ¬ (Atom k) ∈ S ⟶ False)
 ∧ (∀G H. G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S)
 ∧ (∀G H. G ∨ H ∈ S ⟶ G ∈ S ∨ H ∈ S)
 ∧ (∀G H. G → H ∈ S ⟶ ¬G ∈ S ∨ H ∈ S)
 ∧ (∀G. ¬ (¬G) ∈ S ⟶ G ∈ S)
 ∧ (∀G H. ¬(G ∧ H) ∈ S ⟶ ¬ G ∈ S ∨ ¬ H ∈ S)
 ∧ (∀G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S)
 ∧ (∀G H. ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S))
  using assms by (rule auxEq)
 then have C1: ⊥ ∉ S
  by (rule conjunct1)
 have C2: ∀k. Atom k ∈ S ⟶ ¬ (Atom k) ∈ S ⟶ False
  using Hk by (iprover elim: conjunct2 conjunct1)
 have C3: ∀F G H. Con F G H ⟶ F ∈ S ⟶ G ∈ S ∧ H ∈ S
 proof (rule allI)+
  fix F G H
  have C31:∀G H. G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S
   using Hk by (iprover elim: conjunct2 conjunct1)
  have C32:∀G. ¬ (¬ G) ∈ S ⟶ G ∈ S
   using Hk by (iprover elim: conjunct2 conjunct1)
  have C33:∀G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S
   using Hk by (iprover elim: conjunct2 conjunct1)
  have C34:∀G H. ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S
   using Hk by (iprover elim: conjunct2 conjunct1)
  have (∀G H. G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S)
      ∧ (∀G. ¬ (¬ G) ∈ S ⟶ G ∈ S)
      ∧ (∀G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S)
      ∧ (∀G H. ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S)
   using C31 C32 C33 C34 by (iprover intro: conjI)
  thus Con F G H ⟶ F ∈ S ⟶ G ∈ S ∧ H ∈ S
   by (rule Hintikka-alt1Con)
 qed
 have C4:∀F G H. Dis F G H ⟶ F ∈ S ⟶ G ∈ S ∨ H ∈ S
 proof (rule allI)+
```

**fix** $F\ G\ H$
**have** $C41{:}\forall\, G\, H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S$
**using** $Hk$ **by** $(iprover\ elim{:}\ conjunct2\ conjunct1)$
**have** $C42{:}\forall\, G\, H.\ G \rightarrow H \in S \longrightarrow \neg\, G \in S \vee H \in S$
**using** $Hk$ **by** $(iprover\ elim{:}\ conjunct2\ conjunct1)$
**have** $C43{:}\forall\, G.\ \neg\ (\neg\ G) \in S \longrightarrow G \in S$
**using** $Hk$ **by** $(iprover\ elim{:}\ conjunct2\ conjunct1)$
**have** $C44{:}\forall\, G\, H.\ \neg(G \wedge H) \in S \longrightarrow \neg\, G \in S \vee \neg\, H \in S$
**using** $Hk$ **by** $(iprover\ elim{:}\ conjunct2\ conjunct1)$
**have** $(\forall\, G\, H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S)$
$\wedge\ (\forall\, G\, H.\ G \rightarrow H \in S \longrightarrow \neg\, G \in S \vee H \in S)$
$\wedge\ (\forall\, G.\ \neg\ (\neg\ G) \in S \longrightarrow G \in S)$
$\wedge\ (\forall\, G\, H.\ \neg(G \wedge H) \in S \longrightarrow \neg\, G \in S \vee \neg\, H \in S)$
**using** $C41\ C42\ C43\ C44$ **by** $(iprover\ intro{:}\ conjI)$
**thus** $Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \vee H \in S$
**by** $(rule\ Hintikka\text{-}alt1Dis)$
**qed**
**show** $\bot \notin S$
$\wedge\ (\forall\, k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall\, F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall\, F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \vee H \in S)$
**using** $C1\ C2\ C3\ C4$ **by** $(iprover\ intro{:}\ conjI)$
**qed**

Por último, probamos la implicación recíproca de forma detallada en Isabelle mediante el siguiente lema.

**lemma** $Hintikka\text{-}alt2$:
**assumes** $\bot \notin S$
$\wedge\ (\forall\, k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall\, F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall\, F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \vee H \in S)$
**shows** $Hintikka\ S$
**proof** $-$
**have** $Con{:}\forall\, F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \wedge H \in S$
**using** $assms$ **by** $(iprover\ elim{:}\ conjunct2\ conjunct1)$
**have** $Dis{:}\forall\, F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow G \in S \vee H \in S$
**using** $assms$ **by** $(iprover\ elim{:}\ conjunct2\ conjunct1)$
**have** $\bot \notin S$
$\wedge\ (\forall\, k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall\, G\, H.\ G \wedge H \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall\, G\, H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S)$

```
∧ (∀ G H. G → H ∈ S ⟶ ¬G ∈ S ∨ H ∈ S)
∧ (∀ G. ¬ (¬G) ∈ S ⟶ G ∈ S)
∧ (∀ G H. ¬(G ∧ H) ∈ S ⟶ ¬ G ∈ S ∨ ¬ H ∈ S)
∧ (∀ G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S)
∧ (∀ G H. ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S)
proof −
 have C1:⊥ ∉ S
  using assms by (rule conjunct1)
 have C2:∀ k. Atom k ∈ S ⟶ ¬ (Atom k) ∈ S ⟶ False
  using assms by (iprover elim: conjunct2 conjunct1)
 have C3:∀ G H. G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S
 proof (rule allI)+
  fix G H
  show G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S
  proof (rule impI)
   assume G ∧ H ∈ S
   have Con (G ∧ H) G H
    by (simp only: Con.intros(1))
   have Con (G ∧ H) G H ⟶ G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S
    using Con by (iprover elim: allE)
   then have G ∧ H ∈ S ⟶ G ∈ S ∧ H ∈ S
    using ‹Con (G ∧ H) G H› by (rule mp)
   thus G ∈ S ∧ H ∈ S
    using ‹G ∧ H ∈ S› by (rule mp)
  qed
 qed
 have C4:∀ G H. G ∨ H ∈ S ⟶ G ∈ S ∨ H ∈ S
 proof (rule allI)+
  fix G H
  show G ∨ H ∈ S ⟶ G ∈ S ∨ H ∈ S
  proof (rule impI)
   assume G ∨ H ∈ S
   have Dis (G ∨ H) G H
    by (simp only: Dis.intros(1))
   have Dis (G ∨ H) G H ⟶ G ∨ H ∈ S ⟶ G ∈ S ∨ H ∈ S
    using Dis by (iprover elim: allE)
   then have G ∨ H ∈ S ⟶ G ∈ S ∨ H ∈ S
    using ‹Dis (G ∨ H) G H› by (rule mp)
   thus G ∈ S ∨ H ∈ S
    using ‹G ∨ H ∈ S› by (rule mp)
```

**qed**
**qed**
**have** $C5$:$\forall G\, H.\ G \to H \in S \longrightarrow \neg\, G \in S \lor H \in S$
**proof** $(rule\ allI)+$
**fix** $G\ H$
**show** $G \to H \in S \longrightarrow \neg\, G \in S \lor H \in S$
**proof** $(rule\ impI)$
**assume** $G \to H \in S$
**have** $Dis\ (G \to H)\ (\neg\, G)\ H$
**by** $(simp\ only{:}\ Dis.intros(2))$
**have** $Dis\ (G \to H)\ (\neg\, G)\ H \longrightarrow G \to H \in S \longrightarrow \neg\, G \in S \lor H \in S$
**using** $Dis$ **by** $(iprover\ elim{:}\ allE)$
**then have** $G \to H \in S \longrightarrow \neg\, G \in S \lor H \in S$
**using** $\langle Dis\ (G \to H)\ (\neg\, G)\ H\rangle$ **by** $(rule\ mp)$
**thus** $\neg\, G \in S \lor H \in S$
**using** $\langle G \to H \in S\rangle$ **by** $(rule\ mp)$
**qed**
**qed**
**have** $C6$:$\forall G.\ \neg(\neg\, G) \in S \longrightarrow G \in S$
**proof** $(rule\ allI)$
**fix** $G$
**show** $\neg(\neg\, G) \in S \longrightarrow G \in S$
**proof** $(rule\ impI)$
**assume** $\neg\ (\neg\, G) \in S$
**have** $Con\ (\neg\ (\neg\, G))\ G\ G$
**by** $(simp\ only{:}\ Con.intros(4))$
**have** $Con\ (\neg(\neg\, G))\ G\ G \longrightarrow (\neg(\neg\, G)) \in S \longrightarrow G \in S \land G \in S$
**using** $Con$ **by** $(iprover\ elim{:}\ allE)$
**then have** $(\neg(\neg\, G)) \in S \longrightarrow G \in S \land G \in S$
**using** $\langle Con\ (\neg\ (\neg\, G))\ G\ G\rangle$ **by** $(rule\ mp)$
**then have** $G \in S \land G \in S$
**using** $\langle \neg\ (\neg\, G) \in S\rangle$ **by** $(rule\ mp)$
**thus** $G \in S$
**by** $(simp\ only{:}\ conj\text{-}absorb)$
**qed**
**qed**
**have** $C7$:$\forall G\, H.\ \neg(G \land H) \in S \longrightarrow \neg\, G \in S \lor \neg\, H \in S$
**proof** $(rule\ allI)+$
**fix** $G\ H$
**show** $\neg(G \land H) \in S \longrightarrow \neg\, G \in S \lor \neg\, H \in S$

```
    proof (rule impI)
      assume ¬(G ∧ H) ∈ S
      have Dis (¬(G ∧ H)) (¬ G) (¬ H)
        by (simp only: Dis.intros(3))
      have Dis (¬(G ∧ H)) (¬ G) (¬ H) ⟶ ¬(G ∧ H) ∈ S ⟶ ¬ G ∈ S ∨ ¬ H ∈ S
        using Dis by (iprover elim: allE)
      then have ¬(G ∧ H) ∈ S ⟶ ¬ G ∈ S ∨ ¬ H ∈ S
        using ‹Dis (¬(G ∧ H)) (¬ G) (¬ H)› by (rule mp)
      thus ¬ G ∈ S ∨ ¬ H ∈ S
        using ‹¬(G ∧ H) ∈ S› by (rule mp)
    qed
  qed
  have C8:∀ G H. ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S
  proof (rule allI)+
    fix G H
    show ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S
    proof (rule impI)
      assume ¬(G ∨ H) ∈ S
      have Con (¬(G ∨ H)) (¬ G) (¬ H)
        by (simp only: Con.intros(2))
      have Con (¬(G ∨ H)) (¬ G) (¬ H) ⟶ ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S
        using Con by (iprover elim: allE)
      then have ¬(G ∨ H) ∈ S ⟶ ¬ G ∈ S ∧ ¬ H ∈ S
        using ‹Con (¬(G ∨ H)) (¬ G) (¬ H)› by (rule mp)
      thus ¬ G ∈ S ∧ ¬ H ∈ S
        using ‹¬(G ∨ H) ∈ S› by (rule mp)
    qed
  qed
  have C9:∀ G H. ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S
  proof (rule allI)+
    fix G H
    show ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S
    proof (rule impI)
      assume ¬(G → H) ∈ S
      have Con (¬(G → H)) G (¬ H)
        by (simp only: Con.intros(3))
      have Con (¬(G → H)) G (¬ H) ⟶ ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S
        using Con by (iprover elim: allE)
      then have ¬(G → H) ∈ S ⟶ G ∈ S ∧ ¬ H ∈ S
        using ‹Con (¬(G → H)) G (¬ H)› by (rule mp)
```

**thus** $G \in S \wedge \neg\, H \in S$
**using** $\langle\neg(G \rightarrow H) \in S\rangle$ **by** (*rule mp*)
**qed**
**qed**
**have** $A{:}\bot \notin S$
$\wedge\ (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall G\, H.\ G \wedge H \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall G\, H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S)$
$\wedge\ (\forall G\, H.\ G \rightarrow H \in S \longrightarrow \neg G \in S \vee H \in S)$
**using** *C1 C2 C3 C4 C5* **by** (*iprover intro*: *conjI*)
**have** $B{:}(\forall G.\ \neg\ (\neg G) \in S \longrightarrow G \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \wedge H) \in S \longrightarrow \neg\ G \in S \vee \neg\ H \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \vee H) \in S \longrightarrow \neg\ G \in S \wedge \neg\ H \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \rightarrow H) \in S \longrightarrow G \in S \wedge \neg\ H \in S)$
**using** *C6 C7 C8 C9* **by** (*iprover intro*: *conjI*)
**have** $(\bot \notin S$
$\wedge\ (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall G\, H.\ G \wedge H \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall G\, H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S)$
$\wedge\ (\forall G\, H.\ G \rightarrow H \in S \longrightarrow \neg G \in S \vee H \in S))$
$\wedge\ ((\forall G.\ \neg\ (\neg G) \in S \longrightarrow G \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \wedge H) \in S \longrightarrow \neg\ G \in S \vee \neg\ H \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \vee H) \in S \longrightarrow \neg\ G \in S \wedge \neg\ H \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \rightarrow H) \in S \longrightarrow G \in S \wedge \neg\ H \in S))$
**using** *A B* **by** (*rule conjI*)
**thus** $\bot \notin S$
$\wedge\ (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall G\, H.\ G \wedge H \in S \longrightarrow G \in S \wedge H \in S)$
$\wedge\ (\forall G\, H.\ G \vee H \in S \longrightarrow G \in S \vee H \in S)$
$\wedge\ (\forall G\, H.\ G \rightarrow H \in S \longrightarrow \neg G \in S \vee H \in S)$
$\wedge\ (\forall G.\ \neg\ (\neg G) \in S \longrightarrow G \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \wedge H) \in S \longrightarrow \neg\ G \in S \vee \neg\ H \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \vee H) \in S \longrightarrow \neg\ G \in S \wedge \neg\ H \in S)$
$\wedge\ (\forall G\, H.\ \neg(G \rightarrow H) \in S \longrightarrow G \in S \wedge \neg\ H \in S)$
**by** (*iprover intro*: *conj-assoc*)
**qed**
**thus** *Hintikka S*
**unfolding** *Hintikka-def* **by** *this*
**qed**

En conclusión, el lema completo se demuestra detalladamente en Isabelle/HOL como sigue.

**lemma** *Hintikka S* $= (\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow G \in S \wedge H \in S$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow G \in S \vee H \in S$))
**proof** (*rule iffI*)
 **assume** *Hintikka S*
 **thus** ($\bot \notin S$
 $\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
 $\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow G \in S \wedge H \in S$)
 $\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow G \in S \vee H \in S$))
  **by** (*rule Hintikka-alt1*)
**next**
 **assume** ($\bot \notin S$
 $\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
 $\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow G \in S \wedge H \in S$)
 $\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow G \in S \vee H \in S$))
 **thus** *Hintikka S*
  **by** (*rule Hintikka-alt2*)
**qed**

Por último, veamos su demostración automática.

**lemma** *Hintikka-alt*: *Hintikka S* $= (\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow G \in S \wedge H \in S$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow G \in S \vee H \in S$))
 **apply**(*simp add*: *Hintikka-def con-dis-simps*)
 **apply**(*rule iffI*)
  **subgoal by** *blast*
 **subgoal by** *safe metis+*
 **done**

# Capítulo 2

# Propiedad de consistencia proposicional

En este capítulo nos centraremos en demostrar el *teorema de existencia de modelos*. Dicho teorema prueba la satisfacibilidad de un conjunto de fórmulas $S$ si este pertenece a una colección de conjuntos $C$ que verifica la *propiedad de consistencia proposicional*. Para su prueba, definiremos las propiedades de *carácter finito* y *ser cerrada bajo subconjuntos* para colecciones de conjuntos de fórmulas. De este modo, mediante distintos resultados que relacionan estas propiedades con la *propiedad de consistencia proposicional*, dada una colección $C$ cualquiera en las condiciones anteriormente descritas, podemos encontrar una colección $C'$ que la contenga que verifique la *propiedad de consistencia proposicional*, sea *cerrada bajo subconjuntos* y de *carácter finito*. Por otro lado, definiremos una sucesión de conjuntos de fórmulas a partir de la colección $C'$ y el conjunto $S$. Además, definiremos el límite de dicha sucesión que, en particular, contendrá al conjunto $S$. Finalmente probaremos que dicho límite es un conjunto satisfacible por el *lema de Hintikka* y, por contención, quedará probada la satisfacibilidad del conjunto $S$.

En primer lugar, definamos la *propiedad de consistencia proposicional* para una colección de conjuntos de fórmulas proposicionales.

**Definición 2.0.1** *Sea $C$ una colección de conjuntos de fórmulas proposicionales. Decimos que $C$ verifica la propiedad de consistencia proposicional si, para todo conjunto $S$ perteneciente a la colección, se verifica:*

1. $\bot \notin S$.
2. *Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$.*
3. *Si $F \land G \in S$, entonces el conjunto $\{F,G\} \cup S$ pertenece a $C$.*

4. *Si $F \vee G \in S$, entonces o bien el conjunto $\{F\} \cup S$ pertenece a $C$, o bien el conjunto $\{G\}$ $\cup$ $S$ pertenece a $C$.*

5. *Si $F \rightarrow G \in S$, entonces o bien el conjunto $\{\neg F\} \cup S$ pertenece a $C$, o bien el conjunto $\{G\} \cup S$ pertenece a $C$.*

6. *Si $\neg(\neg F) \in S$, entonces el conjunto $\{F\} \cup S$ pertenece a $C$.*

7. *Si $\neg(F \wedge G) \in S$, entonces o bien el conjunto $\{\neg F\} \cup S$ pertenece a $C$, o bien el conjunto $\{\neg G\} \cup S$ pertenece a $C$.*

8. *Si $\neg(F \vee G) \in S$, entonces el conjunto $\{\neg F, \neg G\} \cup S$ pertenece a $C$.*

9. *Si $\neg(F \rightarrow G) \in S$, entonces el conjunto $\{F, \neg G\} \cup S$ pertenece a $C$.*

Veamos, a continuación, su formalización en Isabelle mediante el tipo *definition*.

**definition** $pcp\ C \equiv (\forall S \in C.$
$\bot \notin S$
$\wedge\ (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall F\ G.\ F \wedge G \in S \longrightarrow \{F,G\} \cup S \in C)$
$\wedge\ (\forall F\ G.\ F \vee G \in S \longrightarrow \{F\} \cup S \in C \vee \{G\} \cup S \in C)$
$\wedge\ (\forall F\ G.\ F \rightarrow G \in S \longrightarrow \{\neg F\} \cup S \in C \vee \{G\} \cup S \in C)$
$\wedge\ (\forall F.\ \neg\ (\neg F) \in S \longrightarrow \{F\} \cup S \in C)$
$\wedge\ (\forall F\ G.\ \neg(F \wedge G) \in S \longrightarrow \{\neg F\} \cup S \in C \vee \{\neg G\} \cup S \in C)$
$\wedge\ (\forall F\ G.\ \neg(F \vee G) \in S \longrightarrow \{\neg F, \neg G\} \cup S \in C)$
$\wedge\ (\forall F\ G.\ \neg(F \rightarrow G) \in S \longrightarrow \{F, \neg G\} \cup S \in C))$

Observando la definición anterior, se prueba fácilmente que la colección trivial formada por el conjunto vacío de fórmulas verifica la propiedad de consistencia proposicional.

**lemma** $pcp\ \{\{\}\}$
**unfolding** *pcp-def* **by** *simp*

Del mismo modo, aplicando la definición, se demuestra que los siguientes ejemplos de colecciones de conjuntos de fórmulas proposicionales verifican igualmente la propiedad.

**lemma** $pcp\ \{\{Atom\ 0\}\}$
**unfolding** *pcp-def* **by** *simp*

**lemma** $pcp\ \{\{(\neg\ (Atom\ 1)) \rightarrow Atom\ 2\},$
$\{((\neg\ (Atom\ 1)) \rightarrow Atom\ 2), \neg(\neg\ (Atom\ 1))\},$
$\{((\neg\ (Atom\ 1)) \rightarrow Atom\ 2), \neg(\neg\ (Atom\ 1)),\ Atom\ 1\}\}$

**unfolding** *pcp-def* **by** *auto*

Por último, en contraposición podemos ilustrar un caso de colección que no verifique la propiedad con la siguiente colección obtenida al modificar el último ejemplo. De esta manera, aunque la colección verifique correctamente la quinta condición de la definición, no cumplirá la sexta.

**lemma** $\neg$ *pcp* $\{\{(\neg\ (Atom\ 1)) \rightarrow Atom\ 2\},$
$\{((\neg\ (Atom\ 1)) \rightarrow Atom\ 2), \neg(\neg\ (Atom\ 1))\}\}$
**unfolding** *pcp-def* **by** *auto*

Por otra parte, veamos un resultado que permite la caracterización de la propiedad de consistencia proposicional mediante la notación uniforme.

**Lema 2.0.2 (Caracterización de** *P.C.P* **mediante la notación uniforme)** *Dada una colección C de conjuntos de fórmulas proposicionales, son equivalentes:*

1. *C verifica la propiedad de consistencia proposicional.*
2. *Para cualquier conjunto de fórmulas S de la colección, se verifican las condiciones:*
   - $\bot$ *no pertenece a S.*
   - *Dada p una fórmula atómica cualquiera, no se tiene simultáneamente que* $p \in S$ *y* $\neg\, p \in S$.
   - *Para toda fórmula de tipo* $\alpha$ *con componentes* $\alpha_1$ *y* $\alpha_2$ *tal que* $\alpha$ *pertenece a S, se tiene que* $\{\alpha_1,\alpha_2\} \cup S$ *pertenece a C.*
   - *Para toda fórmula de tipo* $\beta$ *con componentes* $\beta_1$ *y* $\beta_2$ *tal que* $\beta$ *pertenece a S, se tiene que o bien* $\{\beta_1\} \cup S$ *pertenece a C o bien* $\{\beta_2\} \cup S$ *pertenece a C.*

En Isabelle/HOL se formaliza el resultado como sigue.

**lemma** *pcp* $C = (\forall\, S \in C.\ \bot \notin S$
$\wedge\ (\forall\, k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge\ (\forall\, F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C)$
$\wedge\ (\forall\, F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C))$
**oops**

En primer lugar, veamos la demostración del lema.

**Demostración:** Para probar la equivalencia, veamos cada una de las implicaciones por separado.

*1)* $\Longrightarrow$ *2)*

Supongamos que $C$ es una colección de conjuntos de fórmulas proposicionales que verifica la propiedad de consistencia proposicional. Vamos a probar que, en efecto, cumple las condiciones de 2).

Consideremos un conjunto de fórmulas $S$ perteneciente a la colección $C$. Por hipótesis, de la definición de propiedad de consistencia proposicional obtenemos que $S$ verifica las siguientes condiciones:

1. $\perp \notin S$.
2. Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg\, p \in S$.
3. Si $G \wedge H \in S$, entonces el conjunto $\{G,H\} \cup S$ pertenece a $C$.
4. Si $G \vee H \in S$, entonces o bien el conjunto $\{G\} \cup S$ pertenece a $C$, o bien el conjunto $\{H\} \cup S$ pertenece a $C$.
5. Si $G \to H \in S$, entonces o bien el conjunto $\{\neg\, G\} \cup S$ pertenece a $C$, o bien el conjunto $\{H\} \cup S$ pertenece a $C$.
6. Si $\neg(\neg\, G) \in S$, entonces el conjunto $\{G\} \cup S$ pertenece a $C$.
7. Si $\neg(G \wedge H) \in S$, entonces o bien el conjunto $\{\neg\, G\} \cup S$ pertenece a $C$, o bien el conjunto $\{\neg\, H\} \cup S$ pertenece a $C$.
8. Si $\neg(G \vee H) \in S$, entonces el conjunto $\{\neg\, G, \neg\, H\} \cup S$ pertenece a $C$.
9. Si $\neg(G \to H) \in S$, entonces el conjunto $\{G, \neg\, H\} \cup S$ pertenece a $C$.

Las dos primeras condiciones se corresponden con los dos primeros resultados que queríamos demostrar. De este modo, falta probar:

- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$, se tiene que $\{\alpha_1,\alpha_2\} \cup S$ pertenece a $C$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S$, se tiene que o bien $\{\beta_1\} \cup S$ pertenece a $C$ o bien $\{\beta_2\} \cup S$ pertenece a $C$.

En primer lugar, vamos a deducir el primer resultado correspondiente a las fórmulas de tipo $\alpha$ de las condiciones tercera, sexta, octava y novena de la definición de propiedad de consistencia proposicional. En efecto, consideremos una fórmula de tipo $\alpha$ cualquiera con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$. Sabemos que la fórmula es de la forma $G \wedge H$, $\neg\, (\neg\, G)$, $\neg\, (G \vee H)$ o $\neg(G \longrightarrow H)$ para ciertas fórmulas $G$ y $H$. Vamos a probar que para cada caso se cumple que $\{\alpha_1, \alpha_2\} \cup S$ pertenece a la colección:

▪ *Fórmula de tipo* $G \wedge H$: En este caso, sus componentes conjuntivas $\alpha_1$ y $\alpha_2$ son $G$ y $H$ respectivamente. Luego tenemos que $\{\alpha_1, \alpha_2\} \cup S$ pertenece a $C$ por la tercera condición de la definición de propiedad de consistencia proposicional.

▪ *Fórmula de tipo* $\neg\ (\neg\ G)$: En este caso, sus componentes conjuntivas $\alpha_1$ y $\alpha_2$ son ambas $G$. Como el conjunto $\{\alpha_1\} \cup S$ es equivalente a $\{\alpha_1, \alpha_2\} \cup S$ ya que $\alpha_1$ y $\alpha_2$ son iguales, tenemos que este último pertenece a $C$ por la sexta condición de la definición de propiedad de consistencia proposicional.

▪ *Fórmula de tipo* $\neg(G \vee H)$: En este caso, sus componentes conjuntivas $\alpha_1$ y $\alpha_2$ son $\neg\ G$ y $\neg\ H$ respectivamente. Luego tenemos que $\{\alpha_1, \alpha_2\} \cup S$ pertenece a $C$ por la octava condición de la definición de propiedad de consistencia proposicional.

▪ *Fórmula de tipo* $\neg(G \longrightarrow H)$: En este caso, sus componentes conjuntivas $\alpha_1$ y $\alpha_2$ son $G$ y $\neg\ H$ respectivamente. Luego tenemos que $\{\alpha_1, \alpha_2\} \cup S$ pertenece a $C$ por la novena condición de la definición de propiedad de consistencia proposicional.

Finalmente, el resultado correspondiente a las fórmulas de tipo $\beta$ se obtiene de las condiciones cuarta, quinta, sexta y séptima de la definición de propiedad de consistencia proposicional. Para probarlo, consideremos una fórmula cualquiera de tipo $\beta$ perteneciente al conjunto $S$ y cuyas componentes disyuntivas son $\beta_1$ y $\beta_2$. Por simplificación, sabemos que dicha fórmula es de la forma $G \vee H$, $G \longrightarrow H$, $\neg\ (\neg\ G)$ o $\neg(G \wedge H)$ para ciertas fórmulas $G$ y $H$. Deduzcamos que, en efecto, tenemos que o bien $\{\beta_1\} \cup S$ está en $C$ o bien $\{\beta_2\} \cup S$ está en $C$.

▪ *Fórmula de tipo* $G \vee H$: En este caso, sus componentes disyuntivas $\beta_1$ y $\beta_2$ son $G$ y $H$ respectivamente. Luego tenemos que o bien $\{\beta_1\} \cup S$ pertenece a $C$ o bien $\{\beta_2\} \cup S$ pertenece a $C$ por la cuarta condición de la definición de propiedad de consistencia proposicional.

▪ *Fórmula de tipo* $G \longrightarrow H$: En este caso, sus componentes disyuntivas $\beta_1$ y $\beta_2$ son $\neg\ G$ y $H$ respectivamente. Luego tenemos que o bien $\{\beta_1\} \cup S$ pertenece a $C$ o bien $\{\beta_2\} \cup S$ pertenece a $C$ por la quinta condición de la definición de propiedad de consistencia proposicional.

▪ *Fórmula de tipo* $\neg(\neg\ G)$: En este caso, sus componentes disyuntivas $\beta_1$ y $\beta_2$ son ambas $G$. Luego tenemos que, en particular, el conjunto $\{\beta_1\} \cup S$ pertenece a $C$ por la sexta condición de la definición de propiedad de consistencia proposicional. Por tanto, se verifica que o bien $\{\beta_1\} \cup S$ está en $C$ o bien $\{\beta_2\} \cup S$ está en $C$.

▪ *Fórmula de tipo* $\neg(G \wedge H)$: En este caso, sus componentes disyuntivas $\beta_1$ y $\beta_2$ son $\neg\ G$ y $\neg\ H$ respectivamente. Luego tenemos que o bien $\{\beta_1\} \cup S$ pertenece a $C$ o bien $\{\beta_2\} \cup S$ pertenece a $C$ por la séptima condición de la definición de propiedad de consistencia proposicional.

De este modo, queda probada la primera implicación de la equivalencia. Veamos la prueba de la implicación contraria.

*2)* $\Longrightarrow$ *1)*

Supongamos que, dada una colección de conjuntos de fórmulas proposicionales $C$, para cualquier conjunto $S$ de la colección se verifica:

- $\perp$ no pertenece a $S$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$, se tiene que $\{\alpha_1,\alpha_2\} \cup S$ pertenece a $C$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S$, se tiene que o bien $\{\beta_1\} \cup S$ pertenece a $C$ o bien $\{\beta_2\} \cup S$ pertenece a $C$.

Probemos que $C$ verifica la propiedad de consistencia proposicional. Por la definición de la propiedad basta probar que, dado un conjunto cualquiera $S$ perteneciente a $C$, se verifican las siguientes condiciones:

1. $\perp \notin S$.
2. Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$.
3. Si $G \wedge H \in S$, entonces el conjunto $\{G,H\} \cup S$ pertenece a $C$.
4. Si $G \vee H \in S$, entonces o bien el conjunto $\{G\} \cup S$ pertenece a $C$, o bien el conjunto $\{H\} \cup S$ pertenece a $C$.
5. Si $G \to H \in S$, entonces o bien el conjunto $\{\neg G\} \cup S$ pertenece a $C$, o bien el conjunto $\{H\} \cup S$ pertenece a $C$.
6. Si $\neg(\neg G) \in S$, entonces el conjunto $\{G\} \cup S$ pertenece a $C$.
7. Si $\neg(G \wedge H) \in S$, entonces o bien el conjunto $\{\neg G\} \cup S$ pertenece a $C$, o bien el conjunto $\{\neg H\} \cup S$ pertenece a $C$.
8. Si $\neg(G \vee H) \in S$, entonces el conjunto $\{\neg G, \neg H\} \cup S$ pertenece a $C$.
9. Si $\neg(G \to H) \in S$, entonces el conjunto $\{G, \neg H\} \cup S$ pertenece a $C$.

En primer lugar, se observa que por hipótesis se cumplen las dos primeras condiciones de la definición.

Por otra parte, vamos a deducir las condiciones tercera, sexta, octava y novena de la definición de la propiedad de consistencia proposicional a partir de la hipótesis sobre las fórmulas de tipo $\alpha$.

*3*): Supongamos que la fórmula $G \wedge H$ pertenece a $S$ para fórmulas $G$ y $H$ cualesquiera. Observemos que se trata de una fórmula de tipo $\alpha$ de componentes conjuntivas $G$ y $H$. Luego, por hipótesis, tenemos que $\{G, H\} \cup S$ pertenece a $C$.

*6*): Supongamos que la fórmula $\neg(\neg\, G)$ pertenece a $S$ para la fórmula $G$ cualquiera. Observemos que se trata de una fórmula de tipo $\alpha$ cuyas componentes conjuntivas son ambas la fórmula $G$. Por hipótesis, tenemos que el conjunto $\{G,G\} \cup S$ pertence a $C$ y, puesto que dicho conjunto es equivalente a $\{G\} \cup S$, tenemos el resultado.

*8*): Supongamos que la fórmula $\neg(G \vee H)$ pertenece a $S$ para fórmulas $G$ y $H$ cualesquiera. Observemos que se trata de una fórmula de tipo $\alpha$ de componentes conjuntivas $\neg\, G$ y $\neg\, H$. Luego, por hipótesis, tenemos que $\{\neg\, G, \neg\, H\} \cup S$ pertenece a $C$.

*9*): Supongamos que la fórmula $\neg(G \longrightarrow H)$ pertenece a $S$ para fórmulas $G$ y $H$ cualesquiera. Observemos que se trata de una fórmula de tipo $\alpha$ de componentes conjuntivas $G$ y $\neg\, H$. Luego, por hipótesis, tenemos que $\{G, \neg\, H\} \cup S$ pertenece a $C$.

Finalmente, deduzcamos el resto de condiciones de la definición de propiedad de consistencia proposicional a partir de la hipótesis referente a las fórmulas de tipo $\beta$.

*4*): Supongamos que la fórmula $G \vee H$ pertenece a $S$ para fórmulas $G$ y $H$ cualesquiera. Observemos que se trata de una fórmula de tipo $\beta$ de componentes disyuntivas $G$ y $H$. Luego, por hipótesis, tenemos que o bien $\{G\} \cup S$ pertenece a $C$ o bien $\{H\} \cup S$ pertenece a $C$.

*5*): Supongamos que la fórmula $G \longrightarrow H$ pertenece a $S$ para fórmulas $G$ y $H$ cualesquiera. Observemos que se trata de una fórmula de tipo $\beta$ de componentes disyuntivas $\neg\, G$ y $H$. Luego, por hipótesis, tenemos que o bien $\{\neg\, G\} \cup S$ pertenece a $C$ o bien $\{H\} \cup S$ pertenece a $C$.

*7*): Supongamos que la fórmula $\neg(G \wedge H)$ pertenece a $S$ para fórmulas $G$ y $H$ cualesquiera. Observemos que se trata de una fórmula de tipo $\beta$ de componentes disyuntivas $\neg\, G$ y $\neg\, H$. Luego, por hipótesis, tenemos que o bien $\{\neg\, G\} \cup S$ pertenece a $C$ o bien $\{\neg\, H\} \cup S$ pertenece $C$.

De este modo, hemos probado a partir de la hipótesis todas las condiciones que garantizan que la colección $C$ cumple la propiedad de consistencia proposicional. Por lo tanto, queda demostrado el resultado.

□

Análogamente a la demostración del lema anterior de caracterización, para probar este resultado en Isabelle vamos a demostrar cada una de las implicaciones de la equivalencia por separado.

La primera implicación del lema se basa en dos lemas auxiliares. El primero de ellos deduce la condición de 2) sobre fórmulas de tipo $\alpha$ a partir de las condiciones tercera, sexta, octava y novena de la definición de propiedad de consistencia proposicional. Su demostración detallada en Isabelle se muestra a continuación.

```
lemma pcp-alt1Con:
  assumes (∀ G H. G ∧ H ∈ S ⟶ {G,H} ∪ S ∈ C)
  ∧ (∀ G. ¬ (¬G) ∈ S ⟶ {G} ∪ S ∈ C)
  ∧ (∀ G H. ¬(G ∨ H) ∈ S ⟶ {¬ G, ¬ H} ∪ S ∈ C)
  ∧ (∀ G H. ¬(G → H) ∈ S ⟶ {G,¬ H} ∪ S ∈ C)
  shows ∀ F G H. Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C
proof −
  have C1:∀ G H. G ∧ H ∈ S ⟶ {G,H} ∪ S ∈ C
    using assms by (rule conjunct1)
  have C2:∀ G. ¬ (¬G) ∈ S ⟶ {G} ∪ S ∈ C
    using assms by (iprover elim: conjunct2 conjunct1)
  have C3:∀ G H. ¬(G ∨ H) ∈ S ⟶ {¬ G, ¬ H} ∪ S ∈ C
    using assms by (iprover elim: conjunct2 conjunct1)
  have C4:∀ G H. ¬(G → H) ∈ S ⟶ {G,¬ H} ∪ S ∈ C
    using assms by (iprover elim: conjunct2)
  show ∀ F G H. Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C
  proof (rule allI)+
    fix F G H
    show Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C
    proof (rule impI)
      assume Con F G H
      then have F = G ∧ H ∨
           ((∃ G1 H1. F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1) ∨
           (∃ H2. F = ¬ (G → H2) ∧ H = ¬ H2) ∨
           F = ¬ (¬ G) ∧ H = G)
       by (simp only: con-dis-simps(1))
      thus F ∈ S ⟶ {G,H} ∪ S ∈ C
      proof (rule disjE)
```

```
  assume F = G ∧ H
  show F ∈ S ⟶ {G,H} ∪ S ∈ C
    using C1 ‹F = G ∧ H› by (iprover elim: allE)
next
  assume (∃G1 H1. F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1) ∨
        (∃H2. F = ¬ (G → H2) ∧ H = ¬ H2) ∨
        F = ¬ (¬ G) ∧ H = G
  thus F ∈ S ⟶ {G,H} ∪ S ∈ C
  proof (rule disjE)
    assume E1:∃G1 H1. F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1
    obtain G1 H1 where A1:F = ¬ (G1 ∨ H1) ∧ G = ¬ G1 ∧ H = ¬ H1
      using E1 by (iprover elim: exE)
    have F = ¬ (G1 ∨ H1)
      using A1 by (rule conjunct1)
    have G = ¬ G1
      using A1 by (iprover elim: conjunct2 conjunct1)
    have H = ¬ H1
      using A1 by (iprover elim: conjunct2)
    show F ∈ S ⟶ {G,H} ∪ S ∈ C
      using C3 ‹F = ¬ (G1 ∨ H1)› ‹G = ¬ G1› ‹H = ¬ H1› by (iprover elim: allE)
  next
    assume (∃H2. F = ¬ (G → H2) ∧ H = ¬ H2) ∨
          F = ¬ (¬ G) ∧ H = G
    thus F ∈ S ⟶ {G,H} ∪ S ∈ C
    proof (rule disjE)
      assume E2:∃H2. F = ¬ (G → H2) ∧ H = ¬ H2
      obtain H2 where A2:F = ¬ (G → H2) ∧ H = ¬ H2
        using E2 by (rule exE)
      have F = ¬ (G → H2)
        using A2 by (rule conjunct1)
      have H = ¬ H2
        using A2 by (rule conjunct2)
      show F ∈ S ⟶ {G,H} ∪ S ∈ C
        using C4 ‹F = ¬ (G → H2)› ‹H = ¬ H2› by (iprover elim: allE)
    next
      assume A3:F = ¬(¬ G) ∧ H = G
      then have F = ¬(¬ G)
        by (rule conjunct1)
      have H = G
        using A3 by (rule conjunct2)
```

```
        have F ∈ S ⟶ {G} ∪ S ∈ C
          using C2 ‹F = ¬(¬ G)› by (iprover elim: allE)
        then have F ∈ S ⟶ {G,G} ∪ S ∈ C
          by (simp only: insert-absorb2)
        thus F ∈ S ⟶ {G,H} ∪ S ∈ C
          by (simp only: ‹H = G›)
      qed
    qed
   qed
  qed
 qed
qed
```

Finalmente, el siguiente lema auxiliar deduce la condición de 2) sobre fórmulas de tipo $\beta$ a partir de las condiciones cuarta, quinta, sexta y séptima de la definición de propiedad de consistencia proposicional.

```
lemma pcp-alt1Dis:
 assumes (∀ G H. G ∨ H ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C)
 ∧ (∀ G H. G → H ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {H} ∪ S ∈ C)
 ∧ (∀ G. ¬ (¬G) ∈ S ⟶ {G} ∪ S ∈ C)
 ∧ (∀ G H. ¬(G ∧ H) ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {¬ H} ∪ S ∈ C)
 shows ∀ F G H. Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
proof −
 have C1:∀ G H. G ∨ H ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
  using assms by (rule conjunct1)
 have C2:∀ G H. G → H ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
  using assms by (iprover elim: conjunct2 conjunct1)
 have C3:∀ G. ¬ (¬G) ∈ S ⟶ {G} ∪ S ∈ C
  using assms by (iprover elim: conjunct2 conjunct1)
 have C4:∀ G H. ¬(G ∧ H) ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {¬ H} ∪ S ∈ C
  using assms by (iprover elim: conjunct2)
 show ∀ F G H. Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
 proof (rule allI)+
  fix F G H
  show Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
  proof (rule impI)
   assume Dis F G H
   then have F = G ∨ H ∨
        (∃ G1 H1. F = G1 → H1 ∧ G = ¬ G1 ∧ H = H1) ∨
        (∃ G2 H2. F = ¬ (G2 ∧ H2) ∧ G = ¬ G2 ∧ H = ¬ H2) ∨
        F = ¬ (¬ G) ∧ H = G
```

```
  by (simp only: con-dis-simps(2))
thus F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
proof (rule disjE)
  assume F = G ∨ H
  show F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
    using C1 ‹F = G ∨ H› by (iprover elim: allE)
next
  assume (∃ G1 H1. F = G1 → H1 ∧ G = ¬ G1 ∧ H = H1) ∨
      (∃ G2 H2. F = ¬ (G2 ∧ H2) ∧ G = ¬ G2 ∧ H = ¬ H2) ∨
      F = ¬ (¬ G) ∧ H = G
  thus F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
  proof (rule disjE)
    assume E1:∃ G1 H1. F = (G1 → H1) ∧ G = ¬ G1 ∧ H = H1
    obtain G1 H1 where A1: F = (G1 → H1) ∧ G = ¬ G1 ∧ H = H1
      using E1 by (iprover elim: exE)
    have F = (G1 → H1)
      using A1 by (rule conjunct1)
    have G = ¬ G1
      using A1 by (iprover elim: conjunct2 conjunct1)
    have H = H1
      using A1 by (iprover elim: conjunct2)
    show F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
      using C2 ‹F = (G1 → H1)› ‹G = ¬ G1› ‹H = H1› by (iprover elim: allE)
  next
    assume (∃ G2 H2. F = ¬ (G2 ∧ H2) ∧ G = ¬ G2 ∧ H = ¬ H2) ∨
        F = ¬ (¬ G) ∧ H = G
    thus F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
    proof (rule disjE)
      assume E2:∃ G2 H2. F = ¬ (G2 ∧ H2) ∧ G = ¬ G2 ∧ H = ¬ H2
      obtain G2 H2 where A2:F = ¬ (G2 ∧ H2) ∧ G = ¬ G2 ∧ H = ¬ H2
        using E2 by (iprover elim: exE)
      have F = ¬ (G2 ∧ H2)
        using A2 by (rule conjunct1)
      have G = ¬ G2
        using A2 by (iprover elim: conjunct2 conjunct1)
      have H = ¬ H2
        using A2 by (iprover elim: conjunct2)
      show F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
        using C4 ‹F = ¬ (G2 ∧ H2)› ‹G = ¬ G2› ‹H = ¬ H2› by (iprover elim: allE)
    next
```

```
      assume A3:F = ¬(¬ G) ∧ H = G
      then have F = ¬(¬ G)
       by (rule conjunct1)
      have H = G
       using A3 by (rule conjunct2)
      have F ∈ S ⟶ {G} ∪ S ∈ C
       using C3 ‹F = ¬(¬ G)› by (iprover elim: allE)
      then have F ∈ S ⟶ {G} ∪ S ∈ C ∨ {G} ∪ S ∈ C
       by (simp only: disj-absorb)
      thus F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
       by (simp only: ‹H = G›)
     qed
    qed
   qed
  qed
 qed
qed
```

De esta manera, mediante los anteriores lemas auxiliares podemos probar la primera implicación detalladamente en Isabelle.

```
lemma pcp-alt1:
 assumes pcp C
 shows ∀ S ∈ C. ⊥ ∉ S
 ∧ (∀ k. Atom k ∈ S ⟶ ¬ (Atom k) ∈ S ⟶ False)
 ∧ (∀ F G H. Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C)
 ∧ (∀ F G H. Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C)
proof (rule ballI)
 fix S
 assume S ∈ C
 have (∀ S ∈ C.
 ⊥ ∉ S
 ∧ (∀ k. Atom k ∈ S ⟶ ¬ (Atom k) ∈ S ⟶ False)
 ∧ (∀ G H. G ∧ H ∈ S ⟶ {G,H} ∪ S ∈ C)
 ∧ (∀ G H. G ∨ H ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C)
 ∧ (∀ G H. G → H ∈ S ⟶ {¬G} ∪ S ∈ C ∨ {H} ∪ S ∈ C)
 ∧ (∀ G. ¬ (¬G) ∈ S ⟶ {G} ∪ S ∈ C)
 ∧ (∀ G H. ¬(G ∧ H) ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {¬ H} ∪ S ∈ C)
 ∧ (∀ G H. ¬(G ∨ H) ∈ S ⟶ {¬ G, ¬ H} ∪ S ∈ C)
 ∧ (∀ G H. ¬(G → H) ∈ S ⟶ {G,¬ H} ∪ S ∈ C))
  using assms by (simp only: pcp-def)
 then have pcpS:⊥ ∉ S
```

$\wedge$ ($\forall k$. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall G\, H.\ G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ G \vee H \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ G \rightarrow H \in S \longrightarrow \{\neg G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall G.\ \neg\ (\neg G) \in S \longrightarrow \{G\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ \neg(G \wedge H) \in S \longrightarrow \{\neg\, G\} \cup S \in C \vee \{\neg\, H\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ \neg(G \vee H) \in S \longrightarrow \{\neg\, G, \neg\, H\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ \neg(G \rightarrow H) \in S \longrightarrow \{G,\neg\, H\} \cup S \in C$)
 **using** ⟨$S \in C$⟩ **by** (*rule bspec*)
**then have** *C1*:$\bot \notin S$
 **by** (*rule conjunct1*)
**have** *C2*:$\forall k$. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C3*:$\forall G\, H.\ G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C4*:$\forall G\, H.\ G \vee H \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C5*:$\forall G\, H.\ G \rightarrow H \in S \longrightarrow \{\neg G\} \cup S \in C \vee \{H\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C6*:$\forall G.\ \neg\ (\neg G) \in S \longrightarrow \{G\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C7*:$\forall G\, H.\ \neg(G \wedge H) \in S \longrightarrow \{\neg\, G\} \cup S \in C \vee \{\neg\, H\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C8*:$\forall G\, H.\ \neg(G \vee H) \in S \longrightarrow \{\neg\, G, \neg\, H\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** *C9*:$\forall G\, H.\ \neg(G \rightarrow H) \in S \longrightarrow \{G,\neg\, H\} \cup S \in C$
 **using** *pcpS* **by** (*iprover elim*: *conjunct2*)
**have** ($\forall G\, H.\ G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall G.\ \neg\ (\neg G) \in S \longrightarrow \{G\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ \neg(G \vee H) \in S \longrightarrow \{\neg\, G, \neg\, H\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ \neg(G \rightarrow H) \in S \longrightarrow \{G,\neg\, H\} \cup S \in C$)
 **using** *C3 C6 C8 C9* **by** (*iprover intro*: *conjI*)
**then have** *Con*:$\forall F\, G\, H$. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
 **by** (*rule pcp-alt1Con*)
**have** ($\forall G\, H.\ G \vee H \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ G \rightarrow H \in S \longrightarrow \{\neg\, G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall G.\ \neg\ (\neg G) \in S \longrightarrow \{G\} \cup S \in C$)
$\wedge$ ($\forall G\, H.\ \neg(G \wedge H) \in S \longrightarrow \{\neg\, G\} \cup S \in C \vee \{\neg\, H\} \cup S \in C$)
 **using** *C4 C5 C6 C7* **by** (*iprover intro*: *conjI*)
**then have** *Dis*:$\forall F\, G\, H$. *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$

**by** (*rule pcp-alt1Dis*)
**thus** $\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
**using** *C1 C2 Con Dis* **by** (*iprover intro*: *conjI*)
**qed**

Por otro lado, veamos la demostración detallada de la implicación recíproca de la equivalencia. Para ello, utilizaremos distintos lemas auxiliares para deducir cada una de las condiciones de la definición de propiedad de consistencia proposicional a partir de las hipótesis sobre las fórmulas de tipo $\alpha$ y $\beta$. En primer lugar, veamos los lemas que se deducen condiciones a partir de la hipótesis referente a las fórmulas de tipo $\alpha$.

**lemma** *pcp-alt2Con1*:
**assumes** $\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
**shows** $\forall$ *G H*. $G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C$
**proof** (*rule allI*)+
**fix** *G H*
**show** $G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C$
**proof** (*rule impI*)
**assume** $G \wedge H \in S$
**then have** *Con* $(G \wedge H)$ *G H*
**by** (*simp only*: *Con.intros*(*1*))
**let** *?F*=$G \wedge H$
**have** *Con ?F G H* $\longrightarrow$ *?F* $\in S \longrightarrow \{G,H\} \cup S \in C$
**using** *assms* **by** (*iprover elim*: *allE*)
**then have** *?F* $\in S \longrightarrow \{G,H\} \cup S \in C$
**using** ⟨*Con* $(G \wedge H)$ *G H*⟩ **by** (*rule mp*)
**thus** $\{G,H\} \cup S \in C$
**using** ⟨$(G \wedge H) \in S$⟩ **by** (*rule mp*)
**qed**
**qed**

**lemma** *pcp-alt2Con2*:
**assumes** $\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
**shows** $\forall$ *G*. $\neg$ $(\neg G) \in S \longrightarrow \{G\} \cup S \in C$
**proof** (*rule allI*)
**fix** *G*
**show** $\neg$ $(\neg G) \in S \longrightarrow \{G\} \cup S \in C$
**proof** (*rule impI*)
**assume** $\neg(\neg G) \in S$

```
    then have Con (¬(¬G)) G G
      by (simp only: Con.intros(4))
    let ?F=¬(¬ G)
    have ∀ G H. Con ?F G H ⟶ ?F ∈ S ⟶ {G,H} ∪ S ∈ C
      using assms by (rule allE)
    then have ∀ H. Con ?F G H ⟶ ?F ∈ S ⟶ {G,H} ∪ S ∈ C
      by (rule allE)
    then have Con ?F G G ⟶ ?F ∈ S ⟶ {G,G} ∪ S ∈ C
      by (rule allE)
    then have ?F ∈ S ⟶ {G,G} ∪ S ∈ C
      using ‹Con (¬(¬G)) G G› by (rule mp)
    then have {G,G} ∪ S ∈ C
      using ‹(¬(¬G)) ∈ S› by (rule mp)
    thus {G} ∪ S ∈ C
      by (simp only: insert-absorb2)
  qed
qed

lemma pcp-alt2Con3:
  assumes ∀ F G H. Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C
  shows ∀ G H. ¬(G ∨ H) ∈ S ⟶ {¬ G, ¬ H} ∪ S ∈ C
proof (rule allI)+
  fix G H
  show ¬(G ∨ H) ∈ S ⟶ {¬ G, ¬ H} ∪ S ∈ C
  proof (rule impI)
    assume ¬(G ∨ H) ∈ S
    then have Con (¬(G ∨ H)) (¬G) (¬H)
      by (simp only: Con.intros(2))
    let ?F = ¬(G ∨ H)
    have Con ?F (¬G) (¬H) ⟶ ?F ∈ S ⟶ {¬G,¬H} ∪ S ∈ C
      using assms by (iprover elim: allE)
    then have ?F ∈ S ⟶ {¬G,¬H} ∪ S ∈ C
      using ‹Con (¬(G ∨ H)) (¬G) (¬H)› by (rule mp)
    thus {¬G,¬H} ∪ S ∈ C
      using ‹¬(G ∨ H) ∈ S› by (rule mp)
  qed
qed

lemma pcp-alt2Con4:
  assumes ∀ F G H. Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C
```

**shows** $\forall G\ H.\ \neg(G \to H) \in S \longrightarrow \{G,\neg\ H\} \cup S \in C$
**proof** (*rule allI*)+
**fix** *G H*
**show** $\neg(G \to H) \in S \longrightarrow \{G,\neg\ H\} \cup S \in C$
**proof** (*rule impI*)
**assume** $\neg(G \to H) \in S$
**then have** *Con* $(\neg(G \to H))\ G\ (\neg H)$
**by** (*simp only*: *Con.intros*(3))
**let** $?F = \neg(G \to H)$
**have** *Con* $?F\ G\ (\neg H) \longrightarrow ?F \in S \longrightarrow \{G,\neg H\} \cup S \in C$
**using** *assms* **by** (*iprover elim*: *allE*)
**then have** $?F \in S \longrightarrow \{G,\neg H\} \cup S \in C$
**using** ⟨*Con* $(\neg(G \to H))\ G\ (\neg H)$⟩ **by** (*rule mp*)
**thus** $\{G,\neg H\} \cup S \in C$
**using** ⟨$\neg(G \to H) \in S$⟩ **by** (*rule mp*)
**qed**
**qed**

Por otro lado, los siguientes lemas auxiliares prueban el resto de condiciones de la definición de propiedad de consistencia proposicional a partir de la hipótesis referente a fórmulas de tipo $\beta$.

**lemma** *pcp-alt2Dis1*:
**assumes** $\forall F\ G\ H.$ *Dis* $F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**shows** $\forall G\ H.\ G \vee H \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**proof** (*rule allI*)+
**fix** *G H*
**show** $G \vee H \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**proof** (*rule impI*)
**assume** $G \vee H \in S$
**then have** *Dis* $(G \vee H)\ G\ H$
**by** (*simp only*: *Dis.intros*(1))
**let** $?F=G \vee H$
**have** *Dis* $?F\ G\ H \longrightarrow ?F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**using** *assms* **by** (*iprover elim*: *allE*)
**then have** $?F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**using** ⟨*Dis* $(G \vee H)\ G\ H$⟩ **by** (*rule mp*)
**thus** $\{G\} \cup S \in C \vee \{H\} \cup S \in C$
**using** ⟨$(G \vee H) \in S$⟩ **by** (*rule mp*)
**qed**
**qed**

```
lemma pcp-alt2Dis2:
 assumes ∀ F G H. Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
 shows ∀ G H. G → H ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
proof (rule allI)+
 fix G H
 show G → H ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
 proof (rule impI)
  assume G → H ∈ S
  then have Dis (G → H) (¬G) H
   by (simp only: Dis.intros(2))
  let ?F=G → H
  have Dis ?F (¬G) H ⟶ ?F ∈ S ⟶ {¬G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
   using assms by (iprover elim: allE)
  then have ?F ∈ S ⟶ {¬G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
   using ⟨Dis (G → H) (¬G) H⟩ by (rule mp)
  thus {¬G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
   using ⟨(G → H) ∈ S⟩ by (rule mp)
 qed
qed

lemma pcp-alt2Dis3:
 assumes ∀ F G H. Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
 shows ∀ G H. ¬(G ∧ H) ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {¬ H} ∪ S ∈ C
proof (rule allI)+
 fix G H
 show ¬(G ∧ H) ∈ S ⟶ {¬ G} ∪ S ∈ C ∨ {¬ H} ∪ S ∈ C
 proof (rule impI)
  assume ¬(G ∧ H) ∈ S
  then have Dis (¬(G ∧ H)) (¬G) (¬H)
   by (simp only: Dis.intros(3))
  let ?F=¬(G ∧ H)
  have Dis ?F (¬G) (¬H) ⟶ ?F ∈ S ⟶ {¬G} ∪ S ∈ C ∨ {¬H} ∪ S ∈ C
   using assms by (iprover elim: allE)
  then have ?F ∈ S ⟶ {¬G} ∪ S ∈ C ∨ {¬H} ∪ S ∈ C
   using ⟨Dis (¬(G ∧ H)) (¬G) (¬H)⟩ by (rule mp)
  thus {¬G} ∪ S ∈ C ∨ {¬H} ∪ S ∈ C
   using ⟨¬(G ∧ H) ∈ S⟩ by (rule mp)
 qed
qed
```

De este modo, procedemos a la demostración detallada de esta implicación en Isabelle.

**lemma** *pcp-alt2*:
**assumes** $\forall S \in C.\ \bot \notin S$
$\wedge$ $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge$ $(\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C)$
$\wedge$ $(\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C)$
**shows** *pcp C*
**unfolding** *pcp-def*
**proof** (*rule ballI*)
**fix** *S*
**assume** $S \in C$
**have** $H$:$\bot \notin S$
$\wedge$ $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge$ $(\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C)$
$\wedge$ $(\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C)$
**using** *assms* $\langle S \in C \rangle$ **by** (*rule bspec*)
**then have** *Con*:$\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
**by** (*iprover elim*: *conjunct1 conjunct2*)
**have** *Dis*:$\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**using** *H* **by** (*iprover elim*: *conjunct1 conjunct2*)
**have** *1*:$\bot \notin S$
$\wedge$ $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
**using** *H* **by** (*iprover elim*: *conjunct1*)
**have** *2*:$\forall G\ H.\ G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C$
**using** *Con* **by** (*rule pcp-alt2Con1*)
**have** *3*:$\forall G\ H.\ G \vee H \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$
**using** *Dis* **by** (*rule pcp-alt2Dis1*)
**have** *4*:$\forall G\ H.\ G \rightarrow H \in S \longrightarrow \{\neg G\} \cup S \in C \vee \{H\} \cup S \in C$
**using** *Dis* **by** (*rule pcp-alt2Dis2*)
**have** *5*:$\forall G.\ \neg\ (\neg G) \in S \longrightarrow \{G\} \cup S \in C$
**using** *Con* **by** (*rule pcp-alt2Con2*)
**have** *6*:$\forall G\ H.\ \neg(G \wedge H) \in S \longrightarrow \{\neg\ G\} \cup S \in C \vee \{\neg\ H\} \cup S \in C$
**using** *Dis* **by** (*rule pcp-alt2Dis3*)
**have** *7*:$\forall G\ H.\ \neg(G \vee H) \in S \longrightarrow \{\neg\ G, \neg\ H\} \cup S \in C$
**using** *Con* **by** (*rule pcp-alt2Con3*)
**have** *8*:$\forall G\ H.\ \neg(G \rightarrow H) \in S \longrightarrow \{G,\neg\ H\} \cup S \in C$
**using** *Con* **by** (*rule pcp-alt2Con4*)
**have** *A*:$\bot \notin S$
$\wedge$ $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
$\wedge$ $(\forall G\ H.\ G \wedge H \in S \longrightarrow \{G,H\} \cup S \in C)$

$\wedge$ ($\forall$ *G H*. *G* $\vee$ *H* $\in$ *S* $\longrightarrow$ $\{G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. *G* $\rightarrow$ *H* $\in$ *S* $\longrightarrow$ $\{\neg G\} \cup S \in C \vee \{H\} \cup S \in C$)
**using** *1 2 3 4* **by** (*iprover intro*: *conjI*)
**have** *B*:($\forall$ *G*. $\neg$ ($\neg G$) $\in S \longrightarrow \{G\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \wedge H) \in S \longrightarrow \{\neg G\} \cup S \in C \vee \{\neg H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \vee H) \in S \longrightarrow \{\neg G, \neg H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \rightarrow H) \in S \longrightarrow \{G, \neg H\} \cup S \in C$)
**using** *5 6 7 8* **by** (*iprover intro*: *conjI*)
**have** ($\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *G H*. *G* $\wedge$ *H* $\in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. *G* $\vee$ *H* $\in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. *G* $\rightarrow$ *H* $\in S \longrightarrow \{\neg G\} \cup S \in C \vee \{H\} \cup S \in C$))
$\wedge$ (($\forall$ *G*. $\neg$ ($\neg G$) $\in S \longrightarrow \{G\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \wedge H) \in S \longrightarrow \{\neg G\} \cup S \in C \vee \{\neg H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \vee H) \in S \longrightarrow \{\neg G, \neg H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \rightarrow H) \in S \longrightarrow \{G, \neg H\} \cup S \in C$))
**using** *A B* **by** (*rule conjI*)
**thus** $\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *G H*. *G* $\wedge$ *H* $\in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. *G* $\vee$ *H* $\in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. *G* $\rightarrow$ *H* $\in S \longrightarrow \{\neg G\} \cup S \in C \vee \{H\} \cup S \in C$)
$\wedge$ ($\forall$ *G*. $\neg$ ($\neg G$) $\in S \longrightarrow \{G\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \wedge H) \in S \longrightarrow \{\neg G\} \cup S \in C \vee \{\neg H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \vee H) \in S \longrightarrow \{\neg G, \neg H\} \cup S \in C$)
$\wedge$ ($\forall$ *G H*. $\neg(G \rightarrow H) \in S \longrightarrow \{G, \neg H\} \cup S \in C$)
**by** (*iprover intro*: *conj-assoc*)
**qed**

Una vez probadas detalladamente en Isabelle cada una de las implicaciones de la equivalencia, podemos finalmente concluir con la demostración del lema completo.

**lemma** *pcp C* = ($\forall$ *S* $\in$ *C*. $\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$))
**proof** (*rule iffI*)
**assume** *pcp C*
**thus** $\forall$ *S* $\in$ *C*. $\bot \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)

$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G\} \cup S \in C \vee \{H\} \cup S \in C$)
**by** (*rule pcp-alt1*)
**next**
**assume** $\forall$ *S* $\in$ *C*. $\perp \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in$ *S* $\longrightarrow$ $\neg$ (*Atom k*) $\in$ *S* $\longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G,H\} \cup S \in C$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G\} \cup S \in C \vee \{H\} \cup S \in C$)
**thus** *pcp C*
**by** (*rule pcp-alt2*)
**qed**

La demostración automática del resultado en Isabelle/HOL se muestra finalmente a continuación.

**lemma** *pcp-alt*: *pcp C* = ($\forall$ *S* $\in$ *C*.
$\perp \notin S$
$\wedge$ ($\forall$ *k*. *Atom k* $\in$ *S* $\longrightarrow$ $\neg$ (*Atom k*) $\in$ *S* $\longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G,H\} \cup S \in C$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G\} \cup S \in C \vee \{H\} \cup S \in C$))
**apply**(*simp add*: *pcp-def con-dis-simps*)
**apply**(*rule iffI*; *unfold Ball-def*; *elim all-forward*)
**by** (*auto simp add*: *insert-absorb split*: *formula.splits*)

# Capítulo 3

# Colecciones cerradas bajo subconjuntos y colecciones de carácter finito

En este apartado definiremos las propiedades sobre colecciones de conjuntos de ser *cerrada bajo subconjuntos* y de *carácter finito*. Posteriormente daremos distintos resultados que las relacionan con la propiedad de consistencia proposicional y emplearemos en la prueba del *teorema de existencia de modelo*.

**Comentario 5**: Volver a revisar el párrafo anterior al final de la redacción de la sección.

**Definición 3.0.1** *Una colección de conjuntos es cerrada bajo subconjuntos si todo subconjunto de cada conjunto de la colección pertenece a la colección.*

En Isabelle se formaliza de la siguiente manera.

**definition** *subset-closed* $C \equiv (\forall\, S \in C.\ \forall\, S'\subseteq S.\ S' \in C)$

Mostremos algunos ejemplos para ilustrar la definición. Para ello, veamos si las colecciones de conjuntos de fórmulas proposicionales expuestas en los ejemplos anteriores son cerradas bajo subconjuntos.

**lemma** *subset-closed* $\{\{\}\}$
 **unfolding** *subset-closed-def* **by** *simp*

**lemma** $\neg$ *subset-closed* $\{\{Atom\ 0\}\}$
 **unfolding** *subset-closed-def* **by** *auto*

Observemos que, puesto que el conjunto vacío es subconjunto de todo conjunto, una condición necesaria para que una colección sea cerrada bajo subconjuntos es que contenga al conjunto vacío.

**lemma** *subset-closed* $\{\{Atom\ 0\},\{\}\}$
**unfolding** *subset-closed-def* **by** *auto*

De este modo, se deduce fácilmente que el resto de colecciones expuestas en los ejemplos anteriores no son cerradas bajo subconjuntos.

**lemma** $\neg$ *subset-closed* $\{\{(\neg\ (Atom\ 1)) \to Atom\ 2\},$
$\{((\neg\ (Atom\ 1)) \to Atom\ 2), \neg(\neg\ (Atom\ 1))\},$
$\{((\neg\ (Atom\ 1)) \to Atom\ 2), \neg(\neg\ (Atom\ 1)),\ Atom\ 1\}\}$
**unfolding** *subset-closed-def* **by** *auto*

**lemma** $\neg$ *subset-closed* $\{\{(\neg\ (Atom\ 1)) \to Atom\ 2\},$
$\{((\neg\ (Atom\ 1)) \to Atom\ 2), \neg(\neg\ (Atom\ 1))\}\}$
**unfolding** *subset-closed-def* **by** *auto*

Continuemos con la noción de propiedad de carácter finito.

**Definición 3.0.2** *Una colección de conjuntos tiene la propiedad de carácter finito si para cualquier conjunto son equivalentes:*

1. *El conjunto pertenece a la colección.*
2. *Todo subconjunto finito suyo pertenece a la colección.*

La formalización en Isabelle/HOL de dicha definición se muestra a continuación.

**definition** *finite-character* $C \equiv$
$(\forall S.\ S \in C \longleftrightarrow (\forall S' \subseteq S.\ finite\ S' \longrightarrow S' \in C))$

Distingamos las colecciones de los ejemplos anteriores que tengan la propiedad de carácter finito. Análogamente, puesto que el conjunto vacío es finito y subconjunto de cualquier conjunto, se observa que una condición necesaria para que una colección tenga la propiedad de carácter finito es que contenga al conjunto vacío.

**lemma** *finite-character* $\{\{\}\}$
**unfolding** *finite-character-def* **by** *auto*

**lemma** $\neg$ *finite-character* $\{\{Atom\ 0\}\}$
**unfolding** *finite-character-def* **by** *auto*

**lemma** *finite-character* $\{\{Atom\ 0\},\{\}\}$
**unfolding** *finite-character-def* **by** *auto*

Una vez introducidas las definiciones anteriores, veamos los resultados que las relacionan con la propiedad de consistencia proposicional. De este modo, combinándolos en la prueba del *teorema de existencia de modelo*, dada una colección $C$ cualquiera que verifique la propiedad de consistencia proposicional, podemos extenderla a una colección $C'$ que también la verifique y además sea cerrada bajo subconjuntos y de carácter finito.

**Comentario 6**: Volver a revisar el párrafo anterior al final de la redacción de la sección.

**Lema 3.0.3** *Toda colección de conjuntos con la propiedad de consistencia proposicional se puede extender a una colección que también verifique la propiedad de consistencia proposicional y sea cerrada bajo subconjuntos.*

En Isabelle se formaliza el resultado de la siguiente manera.

**lemma** $pcp\ C \Longrightarrow \exists C'.\ C \subseteq C' \land pcp\ C' \land subset\text{-}closed\ C'$
**oops**

Procedamos con su demostración.

**Demostración:** Dada una colección de conjuntos cualquiera $C$, consideremos la colección formada por los conjuntos tales que son subconjuntos de algún conjunto de $C$. Notemos esta colección por $C' = \{S'.\ \exists S{\in}C.\ S' \subseteq S\}$. Vamos a probar que, en efecto, $C'$ verifica las condiciones del lema.

En primer lugar, veamos que $C$ está contenida en $C'$. Para ello, consideremos un conjunto cualquiera perteneciente a $C$. Puesto que la propiedad de contención es reflexiva, dicho conjunto es subconjunto de sí mismo. De este modo, por definición de $C'$, se verifica que el conjunto pertenece a $C'$.

Por otro lado, comprobemos que $C'$ tiene la propiedad de consistencia proposicional. Por el lema de caracterización de la propiedad de consistencia proposicional mediante la notación uniforme basta probar que, para cualquier conjunto de fórmulas $S$ de $C'$, se verifican las condiciones:

- $\bot$ no pertenece a $S$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg\, p \in S$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$, se tiene que $\{\alpha_1,\alpha_2\} \cup S$ pertenece a $C'$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S$, se tiene que o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.

De este modo, sea $S$ un conjunto de fórmulas cualquiera de la colección $C'$. Por definición de dicha colección, existe un conjunto $S'$ pertenciente a $C$ tal que $S$ está contenido en $S'$. Como $C$ tiene la propiedad de consistencia proposicional por hipótesis, verifica las condiciones del lema de caracterización de la propiedad de consistencia proposicional mediante la notación uniforme. En particular, puesto que $S'$ pertenece a $C$, se verifica:

- $\perp$ no pertenece a $S'$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S'$ y $\neg p \in S'$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S'$, se tiene que $\{\alpha_1,\alpha_2\} \cup S'$ pertenece a $C$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S'$, se tiene que o bien $\{\beta_1\} \cup S'$ pertenece a $C$ o bien $\{\beta_2\} \cup S'$ pertenece a $C$.

Por tanto, como $S$ está contenida en $S'$, se verifica análogamente que $\perp$ no pertence a $S$ y que dada una fórmula atómica cualquiera $p$, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$. Veamos que se verifican el resto de condiciones del lema de caracterización:

▪ *Condición para fórmulas de tipo $\alpha$*: Sea una fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$. Como $S$ está contenida en $S'$, tenemos que la fórmula pertence también a $S'$. De este modo, se verifica que $\{\alpha_1,\alpha_2\} \cup S'$ pertenece a la colección $C$. Por otro lado, como el conjunto $S$ está contenido en $S'$, se observa fácilmente que $\{\alpha_1,\alpha_2\} \cup S$ está contenido en $\{\alpha_1,\alpha_2\} \cup S'$. Por lo tanto, el conjunto $\{\alpha_1,\alpha_2\} \cup S$ está en $C'$ por definición de esta, ya que es subconjunto de $\{\alpha_1,\alpha_2\} \cup S'$ que pertence a $C$.

▪ *Condición para fórmulas de tipo $\beta$*: Sea una fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que la fórmula pertenece a $S$. Como el conjunto $S$ está contenido en $S'$, tenemos que la fórmula pertence, a su vez, a $S'$. De este modo, se verifica que o bien $\{\beta_1\} \cup S'$ pertenece a $C$ o bien $\{\beta_2\} \cup S'$ pertence a $C$. Por eliminación de la disyunción anterior, vamos a probar que o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.

- Supongamos, en primer lugar, que $\{\beta_1\} \cup S'$ pertenece a $C$. Puesto que el conjunto $S$ está contenido en $S'$, se observa fácilmente que $\{\beta_1\} \cup S$ está contenido en $\{\beta_1\} \cup S'$. Por definición de la colección $C'$, tenemos que $\{\beta_1\} \cup S$ pertenece a $C'$, ya que es subconjunto de $\{\beta_1\} \cup S'$ que pertenece a $C$. Por tanto, hemos probado que o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.
- Supongamos, finalmente, que $\{\beta_2\} \cup S'$ pertenece a $C$. Análogamente obtenemos que $\{\beta_2\} \cup S$ está contenido en $\{\beta_2\} \cup S'$, luego $\{\beta_2\} \cup S$ pertenece a $C'$ por

definición. Por tanto, o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.

De esta manera, queda probado que dada una fórmula de tipo $\beta$ y componentes $\beta_1$ y $\beta_2$ tal que pertenezca al conjunto $S$, se verifica que o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.

En conclusión, por el lema de caracterización de la propiedad de consistencia proposicional mediante la notación uniforme, queda probado que $C'$ tiene la propiedad de consistencia proposicional.

Finalmente probemos que, además, $C'$ es cerrada bajo subconjuntos. Por definición de ser cerrado bajo subconjuntos, basta probar que dado un conjunto perteneciente a $C'$ verifica que todo subconjunto suyo pertenece a $C'$. Consideremos $S$ un conjunto cualquiera de $C'$. Por definición de $C'$, existe un conjunto $S'$ perteneciente a la colección $C$ tal que $S$ es subconjunto de $S'$. Sea $S''$ un subconjunto cualquiera de $S$. Como $S$ es subconjunto de $S'$, se tiene que $S''$ es, a su vez, subconjunto de $S'$. De este modo, existe un conjunto perteneciente a la colección $C$ del cual $S''$ es subconjunto. Por tanto, por definición de $C'$, $S''$ pertenece a la colección $C'$, como quería demostrar.

$\square$

Procedamos con las demostraciones del lema en Isabelle/HOL.

En primer lugar, vamos a introducir dos lemas auxiliares que emplearemos a lo largo de esta sección. El primero se trata de un lema similar al lema *ballI* definido en Isabelle pero considerando la relación de contención en lugar de la de pertenencia.

**lemma** *sallI*: $(\bigwedge S.\ S \subseteq A \Longrightarrow P\ S) \Longrightarrow \forall S \subseteq A.\ P\ S$
**by** *simp*

Por último definimos el siguiente lema auxiliar similar al lema *bspec* de Isabelle/HOL considerando, análogamente, la relación de contención en lugar de la de pertenencia.

**lemma** *sspec*: $\forall S \subseteq A.\ P\ S \Longrightarrow S \subseteq A \Longrightarrow P\ S$
**by** *simp*

Veamos la prueba detallada del lema en Isabelle/HOL. Esta se fundamenta en tres lemas auxiliares: el primero prueba que la colección $C$ está contenida en $C'$, el segundo que $C'$ tiene la propiedad de consistencia proposicional y, finalmente, el tercer lema demuestra que $C'$ es cerrada bajo subconjuntos. En primer lugar, dada una colección cualquiera $C$, definiremos en Isabelle su extensión $C'$ como sigue.

**definition** *extensionSC* :: $(('a\ formula)\ set)\ set \Rightarrow (('a\ formula)\ set)\ set$
**where** *extensionSC*: $extensionSC\ C = \{s.\ \exists S \in C.\ s \subseteq S\}$

Una vez formalizada la extensión en Isabelle, comencemos probando de manera detallada que toda colección está contenida en su extensión así definida.

**lemma** *ex1-subset*: $C \subseteq$ (*extensionSC C*)
**proof** (*rule subsetI*)
 **fix** *s*
 **assume** $s \in C$
 **have** $s \subseteq s$
  **by** (*rule subset-refl*)
 **then have** $\exists S{\in}C.\ s \subseteq S$
  **using** ⟨$s \in C$⟩ **by** (*rule bexI*)
 **thus** $s \in$ (*extensionSC C*)
  **by** (*simp only*: *mem-Collect-eq extensionSC*)
**qed**

Prosigamos con la prueba del lema auxiliar que demuestra que $C'$ tiene la propiedad de consistencia proposicional. Para ello, emplearemos un lema auxiliar que amplia el lema de Isabelle *insert-is-Un* para la unión de dos elementos y un conjunto, como se muestra a continuación.

**lemma** *insertSetElem*: *insert a* (*insert b C*) $= \{a,b\} \cup C$
 **by** *simp*

Una vez introducido dicho lema auxiliar, podemos dar la prueba detallada del lema que demuestra que $C'$ tiene la propiedad de consistencia proposicional.

**lemma** *ex1-pcp*:
 **assumes** *pcp C*
 **shows** *pcp* (*extensionSC C*)
**proof** −
 **have** *C1*: $C \subseteq$ (*extensionSC C*)
  **by** (*rule ex1-subset*)
 **show** *pcp* (*extensionSC C*)
 **proof** (*rule pcp-alt2*)
  **show** $\forall S \in$ (*extensionSC C*). $(\bot \notin S$
  $\wedge$ ($\forall k$. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
  $\wedge$ ($\forall F\ G\ H$. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in$ (*extensionSC C*))
  $\wedge$ ($\forall F\ G\ H$. *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in$ (*extensionSC C*) $\vee$ $\{H\} \cup S \in$ (*extensionSC C*)))
  **proof** (*rule ballI*)
   **fix** $S'$
   **assume** $S' \in$ (*extensionSC C*)
   **then have** *1*:$\exists S \in C.\ S' \subseteq S$
    **unfolding** *extensionSC* **by** (*rule CollectD*)
   **obtain** *S* **where** $S \in C$ $S' \subseteq S$
    **using** *1* **by** (*rule bexE*)

**have** $\forall S \in C.$
$\bot \notin S$
$\wedge$ ($\forall k.$ *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall F\ G\ H.$ *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall F\ G\ H.$ *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
 **using** *assms* **by** (*rule pcp-alt1*)
**then have** *H*:$\bot \notin S$
$\wedge$ ($\forall k.$ *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall F\ G\ H.$ *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall F\ G\ H.$ *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
 **using** $\langle S \in C \rangle$ **by** (*rule bspec*)
**then have** $\bot \notin S$
 **by** (*rule conjunct1*)
**have** *S1*:$\bot \notin S'$
 **using** $\langle S' \subseteq S \rangle$ $\langle \bot \notin S \rangle$ **by** (*rule contra-subsetD*)
**have** *Atom*:$\forall k.$ *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*
 **using** *H* **by** (*iprover elim*: *conjunct1 conjunct2*)
**have** *S2*:$\forall k.$ *Atom k* $\in S' \longrightarrow \neg$ (*Atom k*) $\in S' \longrightarrow$ *False*
**proof** (*rule allI*)
 **fix** *k*
 **show** *Atom k* $\in S' \longrightarrow \neg$ (*Atom k*) $\in S' \longrightarrow$ *False*
 **proof** (*rule impI*)+
  **assume** *Atom k* $\in S'$
  **assume** $\neg$ (*Atom k*) $\in S'$
  **have** *Atom k* $\in S$
   **using** $\langle S' \subseteq S \rangle$ $\langle$*Atom k* $\in S'\rangle$ **by** (*rule set-mp*)
  **have** $\neg$ (*Atom k*) $\in S$
   **using** $\langle S' \subseteq S \rangle$ $\langle \neg$ (*Atom k*) $\in S'\rangle$ **by** (*rule set-mp*)
  **have** *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*
   **using** *Atom* **by** (*rule allE*)
  **then have** $\neg$ (*Atom k*) $\in S \longrightarrow$ *False*
   **using** $\langle$*Atom k* $\in S\rangle$ **by** (*rule mp*)
  **thus** *False*
   **using** $\langle \neg$ (*Atom k*) $\in S\rangle$ **by** (*rule mp*)
 **qed**
**qed**
**have** *Con*:$\forall F\ G\ H.$ *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
 **using** *H* **by** (*iprover elim*: *conjunct1 conjunct2*)
**have** *S3*:$\forall F\ G\ H.$ *Con F G H* $\longrightarrow F \in S' \longrightarrow \{G,H\} \cup S' \in$ (*extensionSC C*)
**proof** (*rule allI*)+

```
  fix F G H
  show Con F G H ⟶ F ∈ S′ ⟶ {G,H} ∪ S′ ∈ (extensionSC C)
  proof (rule impI)+
   assume Con F G H
   assume F ∈ S′
   have F ∈ S
    using ‹S′ ⊆ S› ‹F ∈ S′› by (rule set-mp)
   have Con F G H ⟶ F ∈ S ⟶ {G,H} ∪ S ∈ C
    using Con by (iprover elim: allE)
   then have F ∈ S ⟶ {G,H} ∪ S ∈ C
    using ‹Con F G H› by (rule mp)
   then have {G,H} ∪ S ∈ C
    using ‹F ∈ S› by (rule mp)
   have S′ ⊆ insert H S
    using ‹S′ ⊆ S› by (rule subset-insertI2)
   then have insert H S′ ⊆ insert H (insert H S)
    by (simp only: insert-mono)
   then have insert H S′ ⊆ insert H S
    by (simp only: insert-absorb2)
   then have insert G (insert H S′) ⊆ insert G (insert H S)
    by (simp only: insert-mono)
   have A:insert G (insert H S′) = {G,H} ∪ S′
    by (rule insertSetElem)
   have B:insert G (insert H S) = {G,H} ∪ S
    by (rule insertSetElem)
   have {G,H} ∪ S′ ⊆ {G,H} ∪ S
    using ‹insert G (insert H S′) ⊆ insert G (insert H S)› by (simp only: A B)
   then have ∃ S ∈ C. {G,H} ∪ S′ ⊆ S
    using ‹{G,H} ∪ S ∈ C› by (rule bexI)
   thus {G,H} ∪ S′ ∈ (extensionSC C)
    unfolding extensionSC by (rule CollectI)
  qed
 qed
 have Dis:∀ F G H. Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
  using H by (iprover elim: conjunct2)
 have S4:∀ F G H. Dis F G H ⟶ F ∈ S′ ⟶ {G} ∪ S′ ∈ (extensionSC C) ∨ {H} ∪ S′ ∈
(extensionSC C)
 proof (rule allI)+
  fix F G H
 show Dis F G H ⟶ F ∈ S′ ⟶ {G} ∪ S′ ∈ (extensionSC C) ∨ {H} ∪ S′ ∈ (extensionSC
```

```
C)
  proof (rule impI)+
   assume Dis F G H
   assume F ∈ S′
   have F ∈ S
    using ‹S′ ⊆ S› ‹F ∈ S′› by (rule set-mp)
   have Dis F G H ⟶ F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
    using Dis by (iprover elim: allE)
   then have F ∈ S ⟶ {G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
    using ‹Dis F G H› by (rule mp)
   then have 9:{G} ∪ S ∈ C ∨ {H} ∪ S ∈ C
    using ‹F ∈ S› by (rule mp)
   show {G} ∪ S′ ∈ (extensionSC C) ∨ {H} ∪ S′ ∈ (extensionSC C)
    using 9
   proof (rule disjE)
    assume {G} ∪ S ∈ C
    have insert G S′ ⊆ insert G S
     using ‹S′ ⊆ S› by (simp only: insert-mono)
    have C:insert G S′ = {G} ∪ S′
     by (rule insert-is-Un)
    have D:insert G S = {G} ∪ S
     by (rule insert-is-Un)
    have {G} ∪ S′ ⊆ {G} ∪ S
     using ‹insert G S′ ⊆ insert G S› by (simp only: C D)
    then have ∃ S ∈ C. {G} ∪ S′ ⊆ S
     using ‹{G} ∪ S ∈ C› by (rule bexI)
    then have {G} ∪ S′ ∈ (extensionSC C)
     unfolding extensionSC by (rule CollectI)
    thus {G} ∪ S′ ∈ (extensionSC C) ∨ {H} ∪ S′ ∈ (extensionSC C)
     by (rule disjI1)
   next
    assume {H} ∪ S ∈ C
    have insert H S′ ⊆ insert H S
     using ‹S′ ⊆ S› by (simp only: insert-mono)
    have E:insert H S′ = {H} ∪ S′
     by (rule insert-is-Un)
    have F:insert H S = {H} ∪ S
     by (rule insert-is-Un)
    then have {H} ∪ S′ ⊆ {H} ∪ S
     using ‹insert H S′ ⊆ insert H S› by (simp only: E F)
```

**then have** $\exists\, S \in C.\ \{H\} \cup S' \subseteq S$
**using** ‹$\{H\} \cup S \in C$› **by** (*rule bexI*)
**then have** $\{H\} \cup S' \in$ (*extensionSC C*)
**unfolding** *extensionSC* **by** (*rule CollectI*)
**thus** $\{G\} \cup S' \in$ (*extensionSC C*) $\vee$ $\{H\} \cup S' \in$ (*extensionSC C*)
**by** (*rule disjI2*)
**qed**
**qed**
**qed**
**show** $\bot \notin S'$
$\wedge$ ($\forall\, k$. *Atom k* $\in S' \longrightarrow \neg$ (*Atom k*) $\in S' \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S' \longrightarrow \{G,H\} \cup S' \in$ (*extensionSC C*))
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S' \longrightarrow \{G\} \cup S' \in$ (*extensionSC C*) $\vee$ $\{H\} \cup S' \in$ (*extensionSC C*))
**using** *S1 S2 S3 S4* **by** (*iprover intro*: *conjI*)
**qed**
**qed**
**qed**

Finalmente, el siguiente lema auxiliar prueba que $C'$ es cerrada bajo subconjuntos.

**lemma** *ex1-subset-closed*:
**assumes** *pcp C*
**shows** *subset-closed* (*extensionSC C*)
**unfolding** *subset-closed-def*
**proof** (*rule ballI*)
**fix** $S'$
**assume** $S' \in$ (*extensionSC C*)
**then have** *H*:$\exists\, S \in C.\ S' \subseteq S$
**unfolding** *extensionSC* **by** (*rule CollectD*)
**obtain** *S* **where** ‹$S \in C$› **and** ‹$S' \subseteq S$›
**using** *H* **by** (*rule bexE*)
**show** $\forall\, S'' \subseteq S'.\ S'' \in$ (*extensionSC C*)
**proof** (*rule sallI*)
**fix** $S''$
**assume** $S'' \subseteq S'$
**then have** $S'' \subseteq S$
**using** ‹$S' \subseteq S$› **by** (*rule subset-trans*)
**then have** $\exists\, S \in C.\ S'' \subseteq S$
**using** ‹$S \in C$› **by** (*rule bexI*)
**thus** $S'' \in$ (*extensionSC C*)
**unfolding** *extensionSC* **by** (*rule CollectI*)

**qed**
**qed**

En conclusión, la prueba detallada del lema completo se muestra a continuación.

**lemma** *ex1*:
**assumes** *pcp C*
**shows** $\exists C'. C \subseteq C' \wedge pcp\ C' \wedge subset\text{-}closed\ C'$
**proof** $-$
**have** *C1*:$C \subseteq (extensionSC\ C)$
**by** (*rule ex1-subset*)
**have** *C2*:*pcp* (*extensionSC C*)
**using** *assms* **by** (*rule ex1-pcp*)
**have** *C3*:*subset-closed* (*extensionSC C*)
**using** *assms* **by** (*rule ex1-subset-closed*)
**have** $C \subseteq (extensionSC\ C) \wedge pcp\ (extensionSC\ C) \wedge subset\text{-}closed\ (extensionSC\ C)$
**using** *C1 C2 C3* **by** (*iprover intro*: *conjI*)
**thus** *?thesis*
**by** (*rule exI*)
**qed**

Continuemos con el segundo resultado de este apartado.

**Lema 3.0.4** *Toda colección de conjuntos con la propiedad de carácter finito es cerrada bajo subconjuntos.*

En Isabelle, se formaliza como sigue.

**lemma**
**assumes** *finite-character C*
**shows** *subset-closed C*
**oops**

Procedamos con la demostración del resultado.

**Demostración:** Consideremos una colección de conjuntos $C$ con la propiedad de carácter finito. Probemos que, en efecto, es cerrada bajo subconjuntos. Por definición de esta última propiedad, basta demostrar que todo subconjunto de cada conjunto de $C$ pertenece también a $C$.

Para ello, tomemos un conjunto $S$ cualquiera perteneciente a $C$ y un subconjunto cualquiera $S'$ de $S$. Probemos que $S'$ está en $C$. Por hipótesis, como $C$ tiene la propiedad de carácter finito, verifica que, para cualquier conjunto $A$, son equivalentes:

1. $A$ pertenece a $C$.

2. Todo subconjunto finito de $A$ pertenece a $C$.

Para probar que el subconjunto $S'$ pertenece a $C$, vamos a demostrar que todo subconjunto finito de $S'$ pertenece a $C$.

De este modo, consideremos un subconjunto cualquiera $S''$ de $S'$. Como $S'$ es subconjunto de $S$, por la transitividad de la relación de contención de conjuntos, se tiene que $S''$ es subconjunto de $S$. Aplicando la definición de propiedad de carácter finito de $C$ para el conjunto $S$, como este pertenece a $C$, verifica que todo subconjunto finito de $S$ pertenece a $C$. En particular, como $S''$ es subconjunto de $S$, verifica que, si $S''$ es finito, entonces $S''$ pertenece a $C$. Por tanto, hemos probado que cualquier conjunto finito de $S'$ pertenece a la colección. Finalmente por la propiedad de carácter finito de $C$, se verifica que $S'$ pertenece a $C$, como queríamos demostrar.

□

Veamos, a continuación, la demostración detallada del resultado en Isabelle.

**lemma**
 **assumes** *finite-character C*
 **shows** *subset-closed C*
 **unfolding** *subset-closed-def*
**proof** (*intro ballI sallI*)
 **fix** $S'$ $S$
 **assume** ‹$S \in C$› **and** ‹$S' \subseteq S$›
 **have** $H$:$\forall A.\ A \in C \longleftrightarrow (\forall A' \subseteq A.\ \mathit{finite}\ A' \longrightarrow A' \in C)$
  **using** *assms* **unfolding** *finite-character-def* **by** *this*
 **have** $QPQ$:$\forall S'' \subseteq S'.\ \mathit{finite}\ S'' \longrightarrow S'' \in C$
 **proof** (*rule sallI*)
  **fix** $S''$
  **assume** $S'' \subseteq S'$
  **then have** $S'' \subseteq S$
   **using** ‹$S' \subseteq S$› **by** (*simp only*: *subset-trans*)
  **have** $1$:$S \in C \longleftrightarrow (\forall S' \subseteq S.\ \mathit{finite}\ S' \longrightarrow S' \in C)$
   **using** $H$ **by** (*rule allE*)
  **have** $\forall S' \subseteq S.\ \mathit{finite}\ S' \longrightarrow S' \in C$
   **using** ‹$S \in C$› $1$ **by** (*rule back-subst*)
  **thus** $\mathit{finite}\ S'' \longrightarrow S'' \in C$
   **using** ‹$S'' \subseteq S$› **by** (*rule sspec*)
 **qed**
 **have** $S' \in C \longleftrightarrow (\forall S'' \subseteq S'.\ \mathit{finite}\ S'' \longrightarrow S'' \in C)$
  **using** $H$ **by** (*rule allE*)

**thus** $S' \in C$
**using** *QPQ* **by** (*rule forw-subst*)
**qed**

Finalmente, su prueba automática en Isabelle/HOL es la siguiente.

**lemma** *ex2*:
**assumes** *fc*: *finite-character C*
**shows** *subset-closed C*
**unfolding** *subset-closed-def*
**proof** (*intro ballI sallI*)
**fix** $S'$ $S$
**assume** *e*: $\langle S \in C\rangle$ **and** *s*: $\langle S' \subseteq S\rangle$
**hence** $*$: $S'' \subseteq S' \Longrightarrow S'' \subseteq S$ **for** $S''$ **by** *simp*
**from** *fc* **have** $S'' \subseteq S \Longrightarrow$ *finite* $S'' \Longrightarrow S'' \in C$ **for** $S''$
**unfolding** *finite-character-def* **using** *e* **by** *blast*
**hence** $S'' \subseteq S' \Longrightarrow$ *finite* $S'' \Longrightarrow S'' \in C$ **for** $S''$ **using** $*$ **by** *simp*
**with** *fc* **show** $\langle S' \in C\rangle$ **unfolding** *finite-character-def* **by** *blast*
**qed**

Introduzcamos el último resultado de la sección.

**Lema 3.0.5** *Toda colección de conjuntos con la propiedad de consistencia proposicional y cerrada bajo subconjuntos se puede extender a una colección que también verifique la propiedad de consistencia proposicional y sea de carácter finito.*

**Demostración:** Dada una colección de conjuntos $C$ en las condiciones del enunciado, vamos a considerar su extensión $C'$ definida como la unión de $C$ y la colección formada por aquellos conjuntos cuyos subconjuntos finitos pertenecen a $C$. Es decir,
$C' = C \cup E$ donde $E = \{S.\ \forall S' \subseteq S.\ \textit{finite}\ S' \longrightarrow S' \in C\}$. Es evidente que es extensión pues contiene a la colección $C$. Vamos a probar que, además es de carácter finito y verifica la propiedad de consistencia proposicional.

En primer lugar, demostremos que $C'$ es de carácter finito. Por definición de dicha propiedad, basta probar que, para cualquier conjunto, son equivalentes:

1. El conjunto pertenece $C'$.

2. Todo subconjunto finito suyo pertenece a $C'$.

Comencemos probando $1) \Longrightarrow 2)$. Para ello, sea un conjunto $S$ de $C'$ de modo que $S'$ es un subconjunto finito suyo. Como $S$ pertenece a la extensión, por definición de la misma tenemos que o bien $S$ está en $C$ o bien $S$ está en $E$. Vamos a probar que $S'$ está en $C'$ por eliminación de la disyunción anterior. En primer lugar, si suponemos que $S$ está en $C$, como se trata de una colección cerrada bajo subconjuntos, tenemos que todo subconjunto de $S$ está en $C$. En particular, $S'$ está en $C$ y, por definición de la extensión, se prueba que $S'$ está en $C'$. Por otro lado, suponiendo que $S$ esté en $E$, por definición de dicha colección tenemos que todo subconjunto finito de $S$ está en $C$. De este modo, por las hipótesis se prueba que $S'$ está en $C$ y, por tanto, pertenece a la extensión.

Por último, probemos la implicación *2*) $\Longrightarrow$ *1*). Sea un conjunto cualquiera $S$ tal que todo subconjunto finito suyo pertenece a $C'$. Vamos a probar que $S$ también pertenece a $C'$. En particular, probaremos que pertenece a $E$. Luego basta probar que todo subconjunto finito de $S$ pertenece a $C$. Para ello, consideremos $S'$ un subconjunto finito cualquiera de $S$. Por hipótesis, tenemos que $S'$ pertenece a $C'$. Por definición de la extensión, tenemos entonces que o bien $S'$ está en $C$ (lo que daría por concluida la prueba) o bien $S'$ está en $E$. De este modo, si suponemos que $S'$ está en $E$, por definición de dicha colección tenemos que todo subconjunto finito suyo está en $C$. En particular, como todo conjunto es subconjunto de si mismo y como hemos supuesto que $S'$ es finito, tenemos que $S'$ está en $C$, lo que prueba la implicación.

Probemos, finalmente, que $C'$ verifica la propiedad de consistencia proposicional. Para ello, vamos a considerar un conjunto cualquiera $S$ perteneciente a $C'$ y probaremos que se verifican las cuatro condiciones del lema de caracterización de la propiedad de consistencia proposicional mediante la notación uniforme. Como el conjunto $S$ pertenece a $C'$, se observa fácilmente por definición de la extensión que, o bien $S$ está en $C$ o bien $S$ está en $E$. Veamos que, para ambos casos, se verifican dichas condiciones.

En primer lugar, supongamos que $S$ está en $C$. Como $C$ verifica la propiedad de consistencia proposicional por hipótesis, verifica el lema de caracterización en particular para el conjunto $S$. De este modo, se cumple:

- $\perp$ no pertenece a $S$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg p \in S$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$, se tiene que $\{\alpha_1, \alpha_2\} \cup S$ pertenece a $C$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S$, se tiene que o bien $\{\beta_1\} \cup S$ pertenece a $C$ o bien $\{\beta_2\} \cup S$ pertenece a $C$.

Por lo tanto, puesto que $C$ está contenida en la extensión $C'$, se verifican las cuatro condiciones del lema para $C'$.

Supongamos ahora que $S$ está en $E$. Probemos que, en efecto, verifica las condiciones del lema de caracterización.

En primer lugar vamos a demostrar que $\perp \notin S$ por reducción al absurdo. Si suponemos que $\perp \in S$, se deduce que el conjunto $\{\perp\}$ es un subconjunto finito de $S$. Como $S$ está en $E$, por definición tenemos que $\{\perp\} \in C$. De este modo, aplicando el lema de caracterización de la propiedad de consistencia proposicional para la colección $C$ y el conjunto $\{\perp\}$, por la primera condición obtenemos que $\perp \notin \{\perp\}$, llegando a una contradicción.

Demostremos que se verifica la segunda condición del lema para las fórmulas atómicas. De este modo, vamos a probar que dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg\, p \in S$. La prueba se realizará por reducción al absurdo, luego supongamos que para cierta fórmula atómica se verifica $p \in S$ y $\neg\, p \in S$. Análogamente, se observa que el conjunto $\{p, \neg\, p\}$ es un subconjunto finito de $S$, luego pertenece a $C$. Aplicando el lema de caracterización de la propiedad de consistencia proposicional para la colección $C$ y el conjunto $\{p, \neg\, p\}$, por la segunda condición obtenemos que no se tiene simultáneamente $q \in \{p, \neg\, p\}$ y $\neg\, q \in \{p, \neg\, p\}$ para ninguna fórmula atómica $q$, llegando así a una contradicción para la fórmula atómica $p$.

Por otro lado, vamos a probar que se verifica la tercera condición del lema de caracterización sobre las fórmulas de tipo $\alpha$. Consideremos una fórmula cualquiera $F$ de tipo $\alpha$ y componentes $\alpha_1$ y $\alpha_2$, y supongamos que $F \in S$. Demostraremos que $\{\alpha_1,\alpha_2\} \cup S \in C'$.

Para ello, probaremos inicialmente que todo subconjunto finito $S'$ de $S$ tal que $F \in S'$ verifica $\{\alpha_1,\alpha_2\} \cup S' \in C$. Consideremos $S'$ subconjunto finito cualquiera de $S$ en las condiciones anteriores. Como $S \in E$, por definición tenemos que $S' \in C$. Aplicando el lema de caracterización de la propiedad de consistencia proposicional para la colección $C$ y el conjunto $S'$, por la tercera condición obtenemos que $\{\alpha_1,\alpha_2\} \cup S' \in C$ ya que hemos supuesto que $F \in S'$.

Una vez probado el resultado anterior, demostremos que $\{\alpha_1,\alpha_2\} \cup S \in E$ y, por definición de $C'$, obtendremos $\{\alpha_1,\alpha_2\} \cup S \in C'$. Además, por definición de $E$, basta probar que todo subconjunto finito de $\{\alpha_1,\alpha_2\} \cup S$ pertenece a $C$. Consideremos $S'$ un subconjunto finito cualquiera de $\{\alpha_1,\alpha_2\} \cup S$. Como $F \in S$, es sencillo comprobar que el conjunto $\{F\} \cup (S' - \{\alpha_1,\alpha_2\})$ es un subconjunto finito de $S$. Por el resultado probado anteriormente, tenemos que el conjunto $\{\alpha_1,\alpha_2\} \cup (\{F\} \cup (S' - \{\alpha_1,\alpha_2\})) =$ $= \{F,\alpha_1,\alpha_2\} \cup S'$ pertenece a $C$. Además, como $C$ es cerrada bajo subconjuntos, todo conjunto de $C$ verifica que cualquier subconjunto suyo pertenece a la colección. Luego, como $S'$ es un subconjunto de $\{F,\alpha_1,\alpha_2\} \cup S'$, queda probado que $S' \in C$.

Finalmente, veamos que se verifica la última condición del lema de caracterización de la propiedad de consistencia proposicional referente a las fórmulas de tipo $\beta$. Consideremos una fórmula cualquiera $F$ de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $F \in S$. Vamos a probar que se tiene que o bien $\{\beta_1\} \cup S \in E$ o bien $\{\beta_1\} \cup S \in E$. En tal caso, por definición de $C'$ se cumple que o bien $\{\beta_1\} \cup S \in C'$ o bien $\{\beta_1\} \cup S \in C'$. La prueba se realizará por reducción al absurdo. Para ello, probemos inicialmente dos resultados previos.

**1**) En las condiciones anteriores, si consideramos $S_1$ y $S_2$ subconjuntos finitos cualesquiera de $S$ tales que $F \in S_1$ y $F \in S_2$, entonces existe una fórmula $I \in \{\beta_1,\beta_2\}$ tal que se verifica que tanto $\{I\} \cup S_1$ como $\{I\} \cup S_2$ están en $C$.

Para probar **1**), consideremos el conjunto finito $S_1 \cup S_2$ que es subconjunto de $S$ por las hipótesis. De este modo, como $S \in E$, tenemos que $S_1 \cup S_2 \in C$. Aplicando el lema de caracterización de la propiedad de consistencia proposicional para la colección $C$ y el conjunto $S_1 \cup S_2$, por la última condición sobre las fórmulas de tipo $\beta$, como $F \in S_1 \cup S_2$ por las hipótesis, se tiene que o bien $\{\beta_1\} \cup S_1 \cup S_2 \in C$ o bien $\{\beta_2\} \cup S_1 \cup S_2 \in C$. Por tanto, existe una fórmula $I \in \{\beta_1,\beta_2\}$ tal que $\{I\} \cup S_1 \cup S_2 \in C$. Sea $I$ la fórmula que cumple lo anterior. Como $C$ es cerrada bajo subconjuntos, los subconjuntos $\{I\} \cup S_1$ y $\{I\} \cup S_2$ de $\{I\} \cup S_1 \cup S_2$ pertenecen también a $C$. Por tanto, hemos probado que existe una fórmula $I \in \{\beta_1,\beta_2\}$ tal que $\{I\} \cup S_1 \in C$ y $\{I\} \cup S_2 \in C$.

Por otra parte, veamos el segundo resultado.

**2**) En las condiciones de **1**) para conjuntos cualesquiera $S_1$ y $S_2$, si además suponemos que $\{\beta_1\} \cup S_1 \notin C$ y $\{\beta_2\} \cup S_2 \notin C$, llegamos a una contradicción.

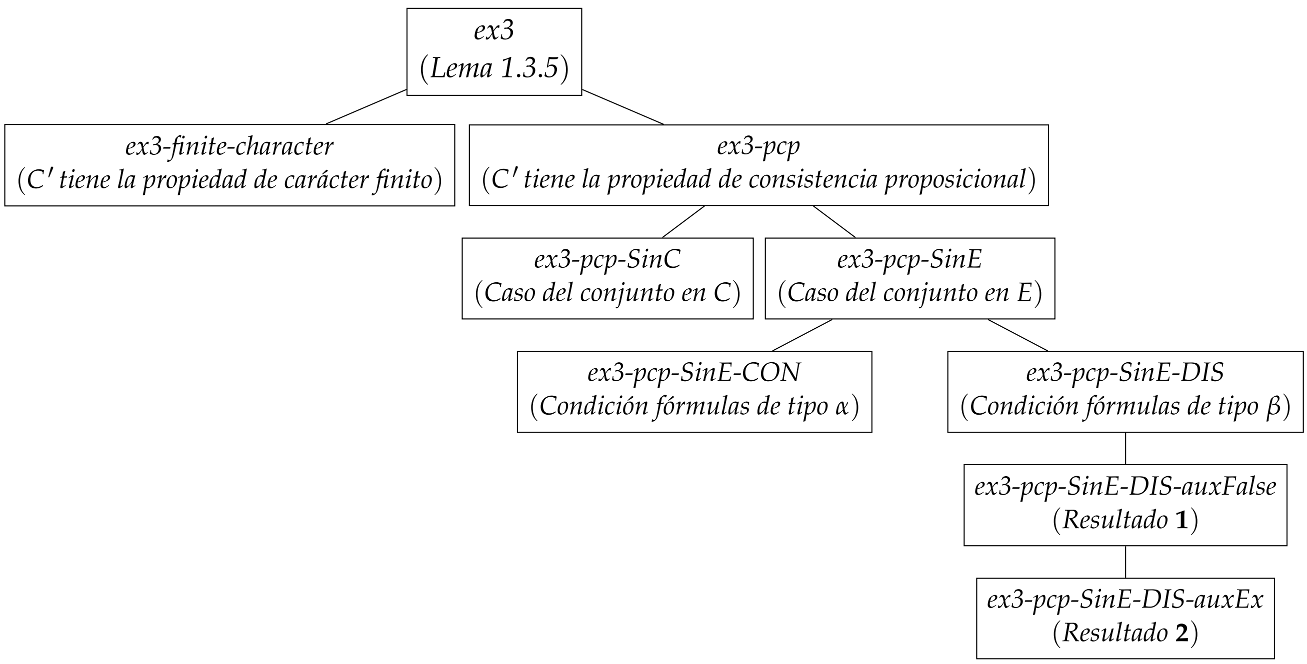
Para probarlo, utilizaremos **1**) para los conjuntos $\{F\} \cup S_1$ y $\{F\} \cup S_2$. Como es evidente, puesto que $F \in S$, se verifica que ambos conjuntos son subconjuntos de $S$. Además, como $S_1$ y $S_2$ son finitos, se tiene que $\{F\} \cup S_1$ y $\{F\} \cup S_2$ también lo son. Por último, es claro que $F$ pertenece a ambos conjuntos. Por lo tanto, por **1**) tenemos que existe una fórmula $I \in \{\beta_1,\beta_2\}$ tal que $\{I\} \cup \{F\} \cup S_1 \in C$ y $\{I\} \cup \{F\} \cup S_2 \in C$. Por otro lado, podemos probar que $\{\beta_1\} \cup \{F\} \cup S_1 \notin C$. Esto se debe a que, en caso contrario, como $C$ es cerrado bajo subconjuntos, tendríamos que el subconjunto $\{\beta_1\} \cup S_1$ pertenecería a $C$, lo que contradice las hipótesis. Análogamente, obtenemos que $\{\beta_2\} \cup \{F\} \cup S_2 \notin C$. De este modo, obtenemos que para toda fórmula $I \in \{\beta_1,\beta_2\}$ se cumple que o bien $\{I\} \cup \{F\} \cup S_1 \notin C$ o bien $\{I\} \cup \{F\} \cup S_2 \notin C$. Esto es equivalente a que no existe ninguna fórmula $I \in \{\beta_1,\beta_2\}$ tal que $\{I\} \cup \{F\} \cup S_1 \in C$ y $\{I\} \cup \{F\} \cup S_2 \in C$, lo que contradice lo obtenido para los conjuntos $\{F\} \cup S_1$ y $\{F\} \cup S_2$ por **1**).

Finalmente, con los resultados anteriores, podemos probar que o bien $\{\beta_1\} \cup S \in E$ o bien $\{\beta_2\} \cup S \in E$ por reducción al absurdo. Supongamos que $\{\beta_1\} \cup S \notin E$ y $\{\beta_2\} \cup S \notin E$. Por definición de $E$, se verifica que existe algún subconjunto finito de $\{\beta_1\} \cup S$ y existe algún subconjunto finito de $\{\beta_2\} \cup S$ tales que no pertenecen a $C$. Notemos por $S_1$ y $S_2$ respectivamente a los subconjuntos anteriores. Vamos a aplicar **2**) para los conjuntos $S_1 - \{\beta_1\}$ y $S_2 - \{\beta_2\}$ para llegar a la contradicción.

Para ello, debemos probar que se verifican las hipótesis del resultado para los conjuntos señalados. Es claro que tanto $S_1 - \{\beta_1\}$ como $S_2 - \{\beta_2\}$ son subconjuntos de $S$, ya que $S_1$ y $S_2$ son subconjuntos de $\{\beta_1\} \cup S$ y $\{\beta_2\} \cup S$ respectivamente. Además, como $S_1$ y $S_2$ son finitos, es evidente que $S_1 - \{\beta_1\}$ y $S_2 - \{\beta_2\}$ también lo son. Queda probar que los conjuntos $\{\beta_1\} \cup (S_1 - \{\beta_1\}) = \{\beta_1\} \cup S_1$ y $\{\beta_2\} \cup (S_2 - \{\beta_2\}) = \{\beta_2\} \cup S_2$ no pertenecen a $C$. Como ni $S_1$ ni $S_2$ están en la colección $C$ cerrada bajo subconjuntos, se cumple que ninguno de ellos son subconjuntos de $S$. Sin embargo, se verifica que $S_1$ es subconjunto de $\{\beta_1\} \cup S$ y $S_2$ es subconjunto de $\{\beta_2\} \cup S$. Por tanto, se cumple que $\beta_1 \in S_1$ y $\beta_2 \in S_2$. Por lo tanto, tenemos finalmente que los conjuntos $\{\beta_1\} \cup S_1 = S_1$ y $\{\beta_2\} \cup S_2 = S_2$ no pertenecen a $C$. Finalmente, como se cumplen las condiciones del resultado 2), llegamos a una contradicción para los conjuntos $S_1 - \{\beta_1\}$ y $S_2 - \{\beta_2\}$, probando que o bien $\{\beta_1\} \cup S \in E$ o bien $\{\beta_1\} \cup S \in E$. Por lo tanto, obtenemos por definición de $C'$ que o bien $\{\beta_1\} \cup S \in C'$ o bien $\{\beta_1\} \cup S \in C'$.

$\square$

Finalmente, veamos la demostración detallada del lema en Isabelle. Debido a la cantidad de lemas auxiliares empleados en la prueba detallada, para facilitar la comprensión mostraremos a continuación un grafo que estructura las relaciones de necesidad de los lemas introducidos.




De este modo, la prueba del *lema 1.3.5* se estructura fundamentalmente en dos lemas auxiliares. El primero, formalizado como *ex3-finite-character* en Isabelle, prueba que la extensión $C' = C \cup E$, donde $E$ es la colección formada por aquellos conjuntos cuyos subconjuntos finitos pertenecen a $C$, tiene la propiedad de carácter finito. El segundo, formalizado como *ex3-pcp*, demuestra que $C'$ verifica la propiedad de consistencia proposicional demostrando que cumple las condiciones suficientes de dicha propiedad por el lema de caracterización *1.2.5*. De este modo, considerando un conjunto $S \in C'$, *ex3-pcp* precisa, a su vez, de dos lemas auxiliares que prueben las condiciones suficientes de *1.2.5*: uno para el caso en que $S \in C$ (*ex3-pcp-SinC*) y otro para el caso en que $S \in E$ (*ex3-pcp-SinE*). Por otro lado, para el último caso en que $S \in E$, utilizaremos dos lemas auxiliares. El primero, formalizado como *ex3-pcp-SinE-CON*, prueba que para $C$ una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ una fórmula de tipo $\alpha$ y componentes $\alpha_1$ y $\alpha_2$, se tiene que $\{\alpha_1,\alpha_2\} \cup S \in C'$. El segundo lema, formalizado como *ex3-pcp-SinE-DIS*, prueba que para $C$ una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ una fórmula de tipo $\beta$ y componentes $\beta_1$ y $\beta_2$, se tiene que o bien $\{\beta_1\} \cup S \in C'$ o bien $\{\beta_2\} \cup S \in C'$. Por último, este segundo lema auxiliar se probará por reducción al absurdo, precisando para ello de los siguientes resultados auxiliares:

*Resultado* **1** Formalizado como *ex3-pcp-SinE-DIS-auxEx*. Prueba que dada $C$ una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ es una fórmula de tipo $\beta$ de componentes $\beta_1$ y $\beta_2$, si consideramos $S_1$ y $S_2$ subconjuntos finitos cualesquiera de $S$ tales que $F \in S_1$ y $F \in S_2$, entonces existe una fórmula $I \in \{\beta_1,\beta_2\}$ tal que se verifica que tanto $\{I\} \cup S_1$ como $\{I\} \cup S_2$ están en $C$.

*Resultado* **2** Formalizado como *ex3-pcp-SinE-DIS-auxFalse*. Utiliza *ex3-pcp-SinE-DIS-auxEx* como lema auxiliar. Prueba que, en las condiciones del *Resultado* **1**, si además suponemos que $\{\beta_1\} \cup S_1 \notin C$ y $\{\beta_2\} \cup S_2 \notin C$, llegamos a una contradicción.

Por otro lado, para facilitar la notación, dada una colección cualquiera $C$, formalizamos las colecciones $E$ y $C'$ como *extF C* y *extensionFin C* respectivamente como se muestra a continuación.

**definition** *extF* :: $(('a\ formula)\ set)\ set \Rightarrow (('a\ formula)\ set)\ set$
**where** *extF*: $extF\ C = \{S.\ \forall S' \subseteq S.\ finite\ S' \longrightarrow S' \in C\}$

**definition** *extensionFin* :: $(('a\ formula)\ set)\ set \Rightarrow (('a\ formula)\ set)\ set$
**where** *extensionFin*: $extensionFin\ C = C \cup (extF\ C)$

Una vez hechas las aclaraciones anteriores, procedamos ordenadamente con la demostración detallada de cada lema auxiliar que conforma la prueba del lema *1.3.5*. En primer lugar, probemos detalladamente que la extensión $C'$ tiene la propiedad de carácter finito.

**lemma** *ex3-finite-character*:

```
 assumes subset-closed C
   shows finite-character (extensionFin C)
proof -
 show finite-character (extensionFin C)
  unfolding finite-character-def
 proof (rule allI)
  fix S
  show S ∈ (extensionFin C) ⟷ (∀S' ⊆ S. finite S' ⟶ S' ∈ (extensionFin C))
  proof (rule iffI)
   assume S ∈ (extensionFin C)
   show ∀S' ⊆ S. finite S' ⟶ S' ∈ (extensionFin C)
   proof (intro sallI impI)
    fix S'
    assume S' ⊆ S
    assume finite S'
    have S ∈ C ∨ S ∈ (extF C)
     using ‹S ∈ (extensionFin C)› by (simp only: extensionFin Un-iff)
    thus S' ∈ (extensionFin C)
    proof (rule disjE)
     assume S ∈ C
     have ∀S ∈ C. ∀S' ⊆ S. S' ∈ C
      using assms by (simp only: subset-closed-def)
     then have ∀S' ⊆ S. S' ∈ C
      using ‹S ∈ C› by (rule bspec)
     then have S' ∈ C
      using ‹S' ⊆ S› by (rule sspec)
     thus S' ∈ (extensionFin C)
      by (simp only: extensionFin UnI1)
    next
     assume S ∈ (extF C)
     then have ∀S' ⊆ S. finite S' ⟶ S' ∈ C
      unfolding extF by (rule CollectD)
     then have finite S' ⟶ S' ∈ C
      using ‹S' ⊆ S› by (rule sspec)
     then have S' ∈ C
      using ‹finite S'› by (rule mp)
     thus S' ∈ (extensionFin C)
      by (simp only: extensionFin UnI1)
    qed
   qed
```

**next**
**assume** $\forall S' \subseteq S.$ *finite* $S' \longrightarrow S' \in$ (*extensionFin* $C$)
**then have** $F$:$\forall S' \subseteq S.$ *finite* $S' \longrightarrow S' \in C \vee S' \in$ (*extF* $C$)
**by** (*simp only*: *extensionFin Un-iff*)
**have** $\forall S' \subseteq S.$ *finite* $S' \longrightarrow S' \in C$
**proof** (*rule sallI*)
**fix** $S'$
**assume** $S' \subseteq S$
**show** *finite* $S' \longrightarrow S' \in C$
**proof** (*rule impI*)
**assume** *finite* $S'$
**have** *finite* $S' \longrightarrow S' \in C \vee S' \in$ (*extF* $C$)
**using** $F$ ⟨$S' \subseteq S$⟩ **by** (*rule sspec*)
**then have** $S' \in C \vee S' \in$ (*extF* $C$)
**using** ⟨*finite* $S'$⟩ **by** (*rule mp*)
**thus** $S' \in C$
**proof** (*rule disjE*)
**assume** $S' \in C$
**thus** $S' \in C$
**by** *this*
**next**
**assume** $S' \in$ (*extF* $C$)
**then have** $S'$:$\forall S'' \subseteq S'.$ *finite* $S'' \longrightarrow S'' \in C$
**unfolding** *extF* **by** (*rule CollectD*)
**have** $S' \subseteq S'$
**by** (*simp only*: *subset-refl*)
**have** *finite* $S' \longrightarrow S' \in C$
**using** $S'$ ⟨$S' \subseteq S'$⟩ **by** (*rule sspec*)
**thus** $S' \in C$
**using** ⟨*finite* $S'$⟩ **by** (*rule mp*)
**qed**
**qed**
**qed**
**then have** $S \in \{S.\ \forall S' \subseteq S.$ *finite* $S' \longrightarrow S' \in C\}$
**by** (*rule CollectI*)
**thus** $S \in$ (*extensionFin* $C$)
**by** (*simp only*: *extF extensionFin UnI2*)
**qed**
**qed**
**qed**

Por otro lado, para probar que $C' = C \cup E$ verifica la propiedad de consistencia proposicional, consideraremos un conjunto $S \in C'$ y utilizaremos fundamentalmente dos lemas auxiliares: uno para el caso en que $S \in C$ y otro para el caso en que $S \in E$.

En primer lugar, vamos a probar el primer lema auxiliar para el caso en que $S \in C$, formalizado como *ex3-pcp-SinC*. Dicho lema prueba que, si $C$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, y sea $S \in C$, se verifican las condiciones del lema de caracterización de la propiedad de consistencia proposicional para la extensión $C'$:

- $\bot \notin S$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg\, p \in S$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$, se tiene que $\{\alpha_1,\alpha_2\} \cup S$ pertenece a $C'$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S$, se tiene que o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.

**lemma** *ex3-pcp-SinC*:
 **assumes** *pcp C*
  *subset-closed C*
  $S \in C$
 **shows** $\bot \notin S \wedge$
  $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False) \wedge$
  $(\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in (extensionFin\ C)) \wedge$
  $(\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in (extensionFin\ C) \vee \{H\} \cup S \in (extensionFin\ C))$
**proof** $-$
 **have** *PCP*:$\forall S \in C.$
  $\bot \notin S$
  $\wedge (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
  $\wedge (\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C)$
  $\wedge (\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C)$
  **using** *assms*(*1*) **by** (*rule pcp-alt1*)
 **have** *H*:$\bot \notin S$
  $\wedge (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
  $\wedge (\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C)$
  $\wedge (\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C)$
  **using** *PCP* ‹$S \in C$› **by** (*rule bspec*)
 **then have** *A1*:$\bot \notin S$

**by** (*rule conjunct1*)
**have** $A2$:$\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False$
**using** $H$ **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** $S3$:$\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
**using** $H$ **by** (*iprover elim*: *conjunct2 conjunct1*)
**have** $A3$:$\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in (extensionFin\ C)$
**proof** (*rule allI*)+
**fix** $F\ G\ H$
**show** $Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in (extensionFin\ C)$
**proof** (*rule impI*)+
**assume** $Con\ F\ G\ H$
**assume** $F \in S$
**have** $Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$
**using** $S3$ **by** (*iprover elim*: *allE*)
**then have** $F \in S \longrightarrow \{G,H\} \cup S \in C$
**using** ⟨$Con\ F\ G\ H$⟩ **by** (*rule mp*)
**then have** $\{G,H\} \cup S \in C$
**using** ⟨$F \in S$⟩ **by** (*rule mp*)
**thus** $\{G,H\} \cup S \in (extensionFin\ C)$
**unfolding** *extensionFin* **by** (*rule UnI1*)
**qed**
**qed**
**have** $S4$:$\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \lor \{H\} \cup S \in C$
**using** $H$ **by** (*iprover elim*: *conjunct2*)
**have** $A4$:$\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in (extensionFin\ C) \lor \{H\} \cup S \in (extensionFin\ C)$
**proof** (*rule allI*)+
**fix** $F\ G\ H$
**show** $Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in (extensionFin\ C) \lor \{H\} \cup S \in (extensionFin\ C)$
**proof** (*rule impI*)+
**assume** $Dis\ F\ G\ H$
**assume** $F \in S$
**have** $Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \lor \{H\} \cup S \in C$
**using** $S4$ **by** (*iprover elim*: *allE*)
**then have** $F \in S \longrightarrow \{G\} \cup S \in C \lor \{H\} \cup S \in C$
**using** ⟨$Dis\ F\ G\ H$⟩ **by** (*rule mp*)
**then have** $\{G\} \cup S \in C \lor \{H\} \cup S \in C$
**using** ⟨$F \in S$⟩ **by** (*rule mp*)
**thus** $\{G\} \cup S \in (extensionFin\ C) \lor \{H\} \cup S \in (extensionFin\ C)$

**proof** (*rule disjE*)
**assume** $\{G\} \cup S \in C$
**then have** $\{G\} \cup S \in$ (*extensionFin* $C$)
**unfolding** *extensionFin* **by** (*rule UnI1*)
**thus** $\{G\} \cup S \in$ (*extensionFin* $C$) $\vee$ $\{H\} \cup S \in$ (*extensionFin* $C$)
**by** (*rule disjI1*)
**next**
**assume** $\{H\} \cup S \in C$
**then have** $\{H\} \cup S \in$ (*extensionFin* $C$)
**unfolding** *extensionFin* **by** (*rule UnI1*)
**thus** $\{G\} \cup S \in$ (*extensionFin* $C$) $\vee$ $\{H\} \cup S \in$ (*extensionFin* $C$)
**by** (*rule disjI2*)
**qed**
**qed**
**qed**
**show** $\bot \notin S \wedge$
($\forall k$. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*) $\wedge$
($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in$ (*extensionFin* $C$)) $\wedge$
($\forall$ *F G H*. *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in$ (*extensionFin* $C$) $\vee$ $\{H\} \cup S \in$ (*extensionFin* $C$))
**using** *A1 A2 A3 A4* **by** (*iprover intro*: *conjI*)
**qed**

Como hemos señalado con anterioridad, para probar el caso en que $S \in E$, donde $E$ es la colección formada por aquellos conjuntos cuyos subconjuntos finitos pertenecen a $C$, precisaremos de distintos lemas auxiliares. El primero de ellos demuestra detalladamente que si $C$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ una fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$, se verifica que $\{\alpha_1,\alpha_2\} \cup S$ pertenece a la extensión $C' = C \cup E$.

**lemma** *ex3-pcp-SinE-CON*:
**assumes** *pcp C*
*subset-closed C*
$S \in$ (*extF C*)
*Con F G H*
$F \in S$
**shows** $\{G,H\} \cup S \in$ (*extensionFin* $C$)
**proof** $-$
**have** *PCP*:$\forall S \in C$.
$\bot \notin S$
$\wedge$ ($\forall k$. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)

$\wedge$ ($\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
  **using** *assms*(1) **by** (*rule pcp-alt1*)
 **have** *1*:$\forall S' \subseteq S.\ finite\ S' \longrightarrow F \in S' \longrightarrow \{G,H\} \cup S' \in C$
 **proof** (*rule sallI*)
  **fix** $S'$
  **assume** $S' \subseteq S$
  **show** $finite\ S' \longrightarrow F \in S' \longrightarrow \{G,H\} \cup S' \in C$
  **proof** (*rule impI*)+
   **assume** $finite\ S'$
   **assume** $F \in S'$
   **have** *E*:$\forall S' \subseteq S.\ finite\ S' \longrightarrow S' \in C$
    **using** *assms*(3) **unfolding** *extF* **by** (*rule CollectD*)
   **then have** $finite\ S' \longrightarrow S' \in C$
    **using** ⟨$S' \subseteq S$⟩ **by** (*rule sspec*)
   **then have** $S' \in C$
    **using** ⟨$finite\ S'$⟩ **by** (*rule mp*)
   **have** $\bot \notin S'$
     $\wedge$ ($\forall k.\ Atom\ k \in S' \longrightarrow \neg\ (Atom\ k) \in S' \longrightarrow False$)
     $\wedge$ ($\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S' \longrightarrow \{G,H\} \cup S' \in C$)
     $\wedge$ ($\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S' \longrightarrow \{G\} \cup S' \in C \vee \{H\} \cup S' \in C$)
    **using** *PCP* ⟨$S' \in C$⟩ **by** (*rule bspec*)
   **then have** $\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S' \longrightarrow \{G, H\} \cup S' \in C$
    **by** (*iprover elim*: *conjunct2 conjunct1*)
   **then have** $Con\ F\ G\ H \longrightarrow F \in S' \longrightarrow \{G, H\} \cup S' \in C$
    **by** (*iprover elim*: *allE*)
   **then have** $F \in S' \longrightarrow \{G,H\} \cup S' \in C$
    **using** *assms*(4) **by** (*rule mp*)
   **thus** $\{G, H\} \cup S' \in C$
    **using** ⟨$F \in S'$⟩ **by** (*rule mp*)
  **qed**
 **qed**
 **have** $\{G,H\} \cup S \in (extF\ C)$
  **unfolding** *mem-Collect-eq Un-iff extF*
 **proof** (*rule sallI*)
  **fix** $S'$
  **assume** *H*:$S' \subseteq \{G,H\} \cup S$
  **show** $finite\ S' \longrightarrow S' \in C$
  **proof** (*rule impI*)
   **assume** $finite\ S'$
   **have** $S' - \{G,H\} \subseteq S$

**using** $H$ **by** $(simp\ only: Diff\text{-}subset\text{-}conv)$
**have** $F \in S \wedge (S' - \{G,H\} \subseteq S)$
**using** $assms(5)$ $\langle S' - \{G,H\} \subseteq S\rangle$ **by** $(rule\ conjI)$
**then have** $insert\ F\ \ (S' - \{G,H\}) \subseteq S$
**by** $(simp\ only: insert\text{-}subset)$
**have** $F1{:}finite\ (insert\ F\ \ (S' - \{G,H\})) \longrightarrow F \in (insert\ F\ \ (S' - \{G,H\})) \longrightarrow \{G,H\} \cup (insert\ F\ \ (S' - \{G,H\})) \in C$
**using** $1$ $\langle insert\ F\ \ (S' - \{G,H\}) \subseteq S\rangle$ **by** $(rule\ sspec)$
**have** $finite\ (S' - \{G,H\})$
**using** $\langle finite\ S'\rangle$ **by** $(rule\ finite\text{-}Diff)$
**then have** $finite\ (insert\ F\ (S' - \{G,H\}))$
**by** $(rule\ finite.insertI)$
**have** $F2{:}F \in (insert\ F\ \ (S' - \{G,H\})) \longrightarrow \{G,H\} \cup (insert\ F\ \ (S' - \{G,H\})) \in C$
**using** $F1$ $\langle finite\ (insert\ F\ (S' - \{G,H\}))\rangle$ **by** $(rule\ mp)$
**have** $F \in (insert\ F\ \ (S' - \{G,H\}))$
**by** $(simp\ only: insertI1)$
**have** $F3{:}\{G,H\} \cup (insert\ F\ (S' - \{G,H\})) \in C$
**using** $F2$ $\langle F \in insert\ F\ (S' - \{G,H\})\rangle$ **by** $(rule\ mp)$
**have** $IU1{:}insert\ F\ (S' - \{G,H\}) = \{F\} \cup (S' - \{G,H\})$
**by** $(rule\ insert\text{-}is\text{-}Un)$
**have** $IU2{:}insert\ F\ (\{G,H\} \cup S') = \{F\} \cup (\{G,H\} \cup S')$
**by** $(rule\ insert\text{-}is\text{-}Un)$
**have** $GH{:}insert\ G\ (insert\ H\ S') = \{G,H\} \cup S'$
**by** $(rule\ insertSetElem)$
**have** $\{G,H\} \cup (insert\ F\ (S' - \{G,H\})) = \{G,H\} \cup (\{F\} \cup (S' - \{G,H\}))$
**by** $(simp\ only: IU1)$
**also have** $\ldots = \{F\} \cup (\{G,H\} \cup (S' - \{G,H\}))$
**by** $(simp\ only: Un\text{-}left\text{-}commute)$
**also have** $\ldots = \{F\} \cup (\{G,H\} \cup S')$
**by** $(simp\ only: Un\text{-}Diff\text{-}cancel)$
**also have** $\ldots = insert\ F\ (\{G,H\} \cup S')$
**by** $(simp\ only: IU2)$
**also have** $\ldots = insert\ F\ (insert\ G\ (insert\ H\ S'))$
**by** $(simp\ only: GH)$
**finally have** $F4{:}\{G,H\} \cup (insert\ F\ (S' - \{G,H\})) = insert\ F\ (insert\ G\ (insert\ H\ S'))$
**by** $this$
**have** $C1{:}insert\ F\ (insert\ G\ (insert\ H\ S')) \in C$
**using** $F3$ **by** $(simp\ only: F4)$
**have** $S' \subseteq insert\ F\ S'$
**by** $(rule\ subset\text{-}insertI)$

**then have** $C2{:}S' \subseteq insert\ F\ (insert\ G\ (insert\ H\ S'))$
  **by** (*simp only*: *subset-insertI2*)
**let** $?S{=}insert\ F\ (insert\ G\ (insert\ H\ S'))$
**have** $\forall\, S \in C.\ \forall\, S' \subseteq S.\ S' \in C$
  **using** *assms*(2) **by** (*simp only*: *subset-closed-def*)
**then have** $\forall\, S' \subseteq ?S.\ S' \in C$
  **using** *C1* **by** (*rule bspec*)
**thus** $S' \in C$
  **using** *C2* **by** (*rule sspec*)
**qed**
**qed**
**thus** $\{G,H\} \cup S \in (extensionFin\ C)$
  **unfolding** *extensionFin* **by** (*rule UnI2*)
**qed**

Seguidamente, vamos a probar el lema auxiliar *ex3-pcp-SinE-DIS*. Este demuestra que si $C$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ una fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$, se verifica que o bien $\{\beta_1\} \cup S \in C'$ o bien $\{\beta_2\} \cup S \in C'$. Dicha prueba se realizará por reducción al absurdo. Para ello precisaremos de dos lemas previos que nos permitan llegar a una contradicción: *ex3-pcp-SinE-DIS-auxEx* y *ex3-pcp-SinE-DIS-auxFalse*.

En primer lugar, veamos la demostración del lema *ex3-pcp-SinE-DIS-auxEx*. Este prueba que dada $C$ una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ es una fórmula de tipo $\beta$ de componentes $\beta_1$ y $\beta_2$, si consideramos $S_1$ y $S_2$ subconjuntos finitos cualesquiera de $S$ tales que $F \in S_1$ y $F \in S_2$, entonces existe una fórmula $I \in \{\beta_1,\beta_2\}$ tal que se verifica que tanto $\{I\} \cup S_1$ como $\{I\} \cup S_2$ están en $C$.

**lemma** *ex3-pcp-SinE-DIS-auxEx*:
**assumes** *pcp C*
  *subset-closed C*
  $S \in (extF\ C)$
  *Dis F G H*
  $S1 \subseteq S$
  *finite S1*
  $F \in S1$
  $S2 \subseteq S$
  *finite S2*
  $F \in S2$
**shows** $\exists\, I{\in}\{G,H\}.\ insert\ I\ S1 \in C \wedge insert\ I\ S2 \in C$
**proof** −

**let** *?S* = *S1* ∪ *S2*
**have** *S1* ⊆ *?S*
**by** (*simp only*: *Un-upper1*)
**have** *S2* ⊆ *?S*
**by** (*simp only*: *Un-upper2*)
**have** *finite ?S*
**using** *assms*(6) *assms*(9) **by** (*rule finite-UnI*)
**have** *?S* ⊆ *S*
**using** *assms*(5) *assms*(8) **by** (*simp only*: *Un-subset-iff*)
**have** ∀ *S'* ⊆ *S*. *finite S'* ⟶ *S'* ∈ *C*
**using** *assms*(3) **unfolding** *extF* **by** (*rule CollectD*)
**then have** *finite ?S* ⟶ *?S* ∈ *C*
**using** ‹*?S* ⊆ *S*› **by** (*rule sspec*)
**then have** *?S* ∈ *C*
**using** ‹*finite ?S*› **by** (*rule mp*)
**have** *F* ∈ *?S*
**using** *assms*(7) **by** (*rule UnI1*)
**have** ∀ *S* ∈ *C*. ⊥ ∉ *S*
∧ (∀ *k*. *Atom k* ∈ *S* ⟶ ¬ (*Atom k*) ∈ *S* ⟶ *False*)
∧ (∀ *F G H*. *Con F G H* ⟶ *F* ∈ *S* ⟶ {*G*,*H*} ∪ *S* ∈ *C*)
∧ (∀ *F G H*. *Dis F G H* ⟶ *F* ∈ *S* ⟶ {*G*} ∪ *S* ∈ *C* ∨ {*H*} ∪ *S* ∈ *C*)
**using** *assms*(1) **by** (*rule pcp-alt1*)
**then have** ⊥ ∉ *?S*
∧ (∀ *k*. *Atom k* ∈ *?S* ⟶ ¬ (*Atom k*) ∈ *?S* ⟶ *False*)
∧ (∀ *F G H*. *Con F G H* ⟶ *F* ∈ *?S* ⟶ {*G*,*H*} ∪ *?S* ∈ *C*)
∧ (∀ *F G H*. *Dis F G H* ⟶ *F* ∈ *?S* ⟶ {*G*} ∪ *?S* ∈ *C* ∨ {*H*} ∪ *?S* ∈ *C*)
**using** ‹*?S* ∈ *C*› **by** (*rule bspec*)
**then have** ∀ *F G H*. *Dis F G H* ⟶ *F* ∈ *?S* ⟶ {*G*} ∪ *?S* ∈ *C* ∨ {*H*} ∪ *?S* ∈ *C*
**by** (*iprover elim*: *conjunct2*)
**then have** *Dis F G H* ⟶ *F* ∈ *?S* ⟶ {*G*} ∪ *?S* ∈ *C* ∨ {*H*} ∪ *?S* ∈ *C*
**by** (*iprover elim*: *allE*)
**then have** *F* ∈ *?S* ⟶ {*G*} ∪ *?S* ∈ *C* ∨ {*H*} ∪ *?S* ∈ *C*
**using** *assms*(4) **by** (*rule mp*)
**then have** *insIsUn*:{*G*} ∪ *?S* ∈ *C* ∨ {*H*} ∪ *?S* ∈ *C*
**using** ‹*F* ∈ *?S*› **by** (*rule mp*)
**have** *insG*:*insert G ?S* = {*G*} ∪ *?S*
**by** (*rule insert-is-Un*)
**have** *insH*:*insert H ?S* = {*H*} ∪ *?S*
**by** (*rule insert-is-Un*)
**have** *insert G ?S* ∈ *C* ∨ *insert H ?S* ∈ *C*

**using** *insG insH* **by** (*simp only*: *insIsUn*)
**then have** (*insert G ?S* $\in$ *C* $\vee$ *insert H ?S* $\in$ *C*) $\vee$ ($\exists$ *I* $\in$ {}. *insert I ?S* $\in$ *C*)
**by** (*simp only*: *disjI1*)
**then have** *insert G ?S* $\in$ *C* $\vee$ (*insert H ?S* $\in$ *C* $\vee$ ($\exists$ *I* $\in$ {}. *insert I ?S* $\in$ *C*))
**by** (*simp only*: *disj-assoc*)
**then have** *insert G ?S* $\in$ *C* $\vee$ ($\exists$ *I* $\in$ {*H*}. *insert I ?S* $\in$ *C*)
**by** (*simp only*: *bex-simps*(*5*))
**then have** *1*:$\exists$ *I* $\in$ {*G*,*H*}. *insert I ?S* $\in$ *C*
**by** (*simp only*: *bex-simps*(*5*))
**obtain** *I* **where** *I* $\in$ {*G*,*H*} **and** *insert I ?S* $\in$ *C*
**using** *1* **by** (*rule bexE*)
**have** *SC*:$\forall$ *S* $\in$ *C*. $\forall$ *S'*$\subseteq$*S*. *S'* $\in$ *C*
**using** *assms*(*2*) **by** (*simp only*: *subset-closed-def*)
**then have** *2*:$\forall$ *S'* $\subseteq$ (*insert I ?S*). *S'* $\in$ *C*
**using** ⟨*insert I ?S* $\in$ *C*⟩ **by** (*rule bspec*)
**have** *insert I S1* $\subseteq$ *insert I ?S*
**using** ⟨*S1* $\subseteq$ *?S*⟩ **by** (*rule insert-mono*)
**have** *insert I S1* $\in$ *C*
**using** *2* ⟨*insert I S1* $\subseteq$ *insert I ?S*⟩ **by** (*rule sspec*)
**have** *insert I S2* $\subseteq$ *insert I ?S*
**using** ⟨*S2* $\subseteq$ *?S*⟩ **by** (*rule insert-mono*)
**have** *insert I S2* $\in$ *C*
**using** *2* ⟨*insert I S2* $\subseteq$ *insert I ?S*⟩ **by** (*rule sspec*)
**have** *insert I S1* $\in$ *C* $\wedge$ *insert I S2* $\in$ *C*
**using** ⟨*insert I S1* $\in$ *C*⟩ ⟨*insert I S2* $\in$ *C*⟩ **by** (*rule conjI*)
**thus** $\exists$ *I*$\in${*G*,*H*}. *insert I S1* $\in$ *C* $\wedge$ *insert I S2* $\in$ *C*
**using** ⟨*I* $\in$ {*G*,*H*}⟩ **by** (*rule bexI*)
**qed**

Finalmente, el lema *ex3-pcp-SinE-DIS-auxFalse* prueba que dada una colección $C$ con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ es una fórmula de tipo $\beta$ de componentes $\beta_1$ y $\beta_2$, si consideramos $S_1$ y $S_2$ subconjuntos finitos cualesquiera de $S$ tales que $F \in S_1$, $F \in S_2$, $\{\beta_1\} \cup S_1 \notin C$ y $\{\beta_2\} \cup S_2 \notin C$, llegamos a una contradicción.

**lemma** *ex3-pcp-SinE-DIS-auxFalse*:
**assumes** *pcp C*
*subset-closed C*
*S* $\in$ (*extF C*)
*Dis F G H*
*F* $\in$ *S*
*S1* $\subseteq$ *S*

*finite S1*
*insert G S1 ∉ C*
*S2 ⊆ S*
*finite S2*
*insert H S2 ∉ C*
**shows** *False*
**proof** −
**let** *?S1=insert F S1*
**let** *?S2=insert F S2*
**have** *SC:∀ S ∈ C. ∀ S′⊆S. S′ ∈ C*
**using** *assms*(2) **by** (*simp only*: *subset-closed-def*)
**have** *1:?S1 ⊆ S*
**using** ⟨*F ∈ S*⟩ ⟨*S1 ⊆ S*⟩ **by** (*simp only*: *insert-subset*)
**have** *2:finite ?S1*
**using** ⟨*finite S1*⟩ **by** (*simp only*: *finite-insert*)
**have** *3:F ∈ ?S1*
**by** (*simp only*: *insertI1*)
**have** *4:insert G ?S1 ∉ C*
**proof** (*rule ccontr*)
**assume** ¬(*insert G ?S1 ∉ C*)
**then have** *insert G ?S1 ∈ C*
**by** (*rule notnotD*)
**have** *SC1:∀ S′ ⊆* (*insert G ?S1*). *S′ ∈ C*
**using** *SC* ⟨*insert G ?S1 ∈ C*⟩ **by** (*rule bspec*)
**have** *insert G S1 ⊆ insert F* (*insert G S1*)
**by** (*rule subset-insertI*)
**then have** *insert G S1 ⊆ insert G ?S1*
**by** (*simp only*: *insert-commute*)
**have** *insert G S1 ∈ C*
**using** *SC1* ⟨*insert G S1 ⊆ insert G ?S1*⟩ **by** (*rule sspec*)
**show** *False*
**using** *assms*(*8*) ⟨*insert G S1 ∈ C*⟩ **by** (*rule notE*)
**qed**
**have** *5:?S2 ⊆ S*
**using** ⟨*F ∈ S*⟩ ⟨*S2 ⊆ S*⟩ **by** (*simp only*: *insert-subset*)
**have** *6:finite ?S2*
**using** ⟨*finite S2*⟩ **by** (*simp only*: *finite-insert*)
**have** *7:F ∈ ?S2*
**by** (*simp only*: *insertI1*)
**have** *8:insert H ?S2 ∉ C*

**proof** (*rule ccontr*)
  **assume** $\neg$(*insert H ?S2* $\notin$ *C*)
  **then have** *insert H ?S2* $\in$ *C*
    **by** (*rule notnotD*)
  **have** *SC2*:$\forall$ *S'* $\subseteq$ (*insert H ?S2*). *S'* $\in$ *C*
    **using** *SC* ⟨*insert H ?S2* $\in$ *C*⟩ **by** (*rule bspec*)
  **have** *insert H S2* $\subseteq$ *insert F* (*insert H S2*)
    **by** (*rule subset-insertI*)
  **then have** *insert H S2* $\subseteq$ *insert H ?S2*
    **by** (*simp only*: *insert-commute*)
  **have** *insert H S2* $\in$ *C*
    **using** *SC2* ⟨*insert H S2* $\subseteq$ *insert H ?S2*⟩ **by** (*rule sspec*)
  **show** *False*
    **using** *assms*(*11*) ⟨*insert H S2* $\in$ *C*⟩ **by** (*rule notE*)
 **qed**
 **have** *Ex*:$\exists$ *I* $\in$ {*G*,*H*}. *insert I ?S1* $\in$ *C* $\wedge$ *insert I ?S2* $\in$ *C*
  **using** *assms*(*1*) *assms*(*2*) *assms*(*3*) *assms*(*4*) *1 2 3 5 6 7* **by** (*rule ex3-pcp-SinE-DIS-auxEx*)
 **have** $\forall$ *I* $\in$ {*G*,*H*}. *insert I ?S1* $\notin$ *C* $\vee$ *insert I ?S2* $\notin$ *C*
  **using** *4 8* **by** *simp*
 **then have** $\forall$ *I* $\in$ {*G*,*H*}. $\neg$(*insert I ?S1* $\in$ *C* $\wedge$ *insert I ?S2* $\in$ *C*)
  **by** (*simp only*: *de-Morgan-conj*)
 **then have** $\neg$($\exists$ *I* $\in$ {*G*,*H*}. *insert I ?S1* $\in$ *C* $\wedge$ *insert I ?S2* $\in$ *C*)
  **by** (*simp only*: *bex-simps*(*8*))
 **thus** *False*
  **using** *Ex* **by** (*rule notE*)
**qed**

Una vez introducidos los lemas anteriores, podemos probar el lema *ex3-pcp-SinE-DIS* que demuestra que si $C$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, $S \in E$ y sea $F$ una fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$, se verifica que o bien $\{\beta_1\} \cup S \in C'$ o bien $\{\beta_2\} \cup S \in C'$. Además, para dicha prueba necesitaremos los siguientes lemas auxiliares en Isabelle.

**lemma** *sall-simps-not-all*:
 **assumes** $\neg(\forall\, x \subseteq A.\ P\ x)$
 **shows** $\exists\, x \subseteq A.\ (\neg\ P\ x)$
 **using** *assms* **by** *blast*

**lemma** *subexE*: $\exists\, x{\subseteq}A.\ P\ x \Longrightarrow (\bigwedge x.\ x{\subseteq}A \Longrightarrow P\ x \Longrightarrow Q) \Longrightarrow Q$
 **by** *blast*

De este modo, procedamos con la demostración detallada de *ex3-pcp-SinE-DIS*.

**lemma** *ex3-pcp-SinE-DIS*:
  **assumes** *pcp C*
        *subset-closed C*
        $S \in (extF\ C)$
        *Dis F G H*
        $F \in S$
  **shows** $\{G\} \cup S \in (extensionFin\ C) \vee \{H\} \cup S \in (extensionFin\ C)$
**proof** −
  **have** $(extF\ C) \subseteq (extensionFin\ C)$
    **unfolding** *extensionFin* **by** (*rule Un-upper2*)
  **have** *PCP*:$\forall S \in C.$
        $\perp \notin S$
        $\wedge (\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)$
        $\wedge (\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G{,}H\} \cup S \in C)$
        $\wedge (\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C)$
    **using** *assms*(*1*) **by** (*rule pcp-alt1*)
  **have** *E*:$\forall S' \subseteq S.\ finite\ S' \longrightarrow S' \in C$
    **using** *assms*(*3*) **unfolding** *extF* **by** (*rule CollectD*)
  **then have** *E'*:$\forall S'.\ S' \subseteq S \longrightarrow finite\ S' \longrightarrow S' \in C$
    **by** *blast*
  **have** *SC*:$\forall S \in C.\ \forall S' \subseteq S.\ S' \in C$
    **using** *assms*(*2*) **by** (*simp only*: *subset-closed-def*)
  **have** $insert\ G\ S \in (extF\ C) \vee insert\ H\ S \in (extF\ C)$
  **proof** (*rule ccontr*)
    **assume** $\neg(insert\ G\ S \in (extF\ C) \vee insert\ H\ S \in (extF\ C))$
    **then have** *Conj*:$\neg(insert\ G\ S \in (extF\ C)) \wedge \neg(insert\ H\ S \in (extF\ C))$
      **by** (*simp only*: *simp-thms*(*8*,*25*) *de-Morgan-disj*)
    **then have** $\neg(insert\ G\ S \in (extF\ C))$
      **by** (*rule conjunct1*)
    **then have** $\neg(\forall S' \subseteq (insert\ G\ S).\ finite\ S' \longrightarrow S' \in C)$
      **unfolding** *extF* **by** (*simp add*: *mem-Collect-eq*)
    **then have** *Ex1*:$\exists S' \subseteq (insert\ G\ S).\ \neg(finite\ S' \longrightarrow S' \in C)$
      **by** (*rule sall-simps-not-all*)
    **obtain** *S1* **where** $S1 \subseteq insert\ G\ S$ **and** $\neg(finite\ S1 \longrightarrow S1 \in C)$
      **using** *Ex1* **by** (*rule subexE*)
    **have** $finite\ S1 \wedge S1 \notin C$
      **using** $\langle\neg(finite\ S1 \longrightarrow S1 \in C)\rangle$ **by** (*simp only*: *simp-thms*(*8*) *not-imp*)
    **then have** *finite S1*
      **by** (*rule conjunct1*)
    **have** $S1 \notin C$

**using** $\langle finite\ S1 \wedge S1 \notin C\rangle$ **by** $(rule\ conjunct2)$
**then have** $insert\ G\ S1 \notin C$
**proof** $-$
**have** $S1 \subseteq S \longrightarrow finite\ S1 \longrightarrow S1 \in C$
**using** $E'$ **by** $(rule\ allE)$
**then have** $\neg\ S1 \subseteq S$
**using** $\langle\neg\ (finite\ S1 \longrightarrow S1 \in C)\rangle$ **by** $(rule\ mt)$
**then have** $(S1 \subseteq insert\ G\ S) \neq (S1 \subseteq S)$
**using** $\langle S1 \subseteq insert\ G\ S\rangle$ **by** *simp*
**then have** $notSI$:$\neg(S1 \subseteq insert\ G\ S \longleftrightarrow S1 \subseteq S)$
**by** *blast*
**have** $subsetInsert$:$G \notin S1 \Longrightarrow S1 \subseteq insert\ G\ S \longleftrightarrow S1 \subseteq S$
**by** $(rule\ subset\text{-}insert)$
**have** $\neg(G \notin S1)$
**using** *notSI subsetInsert* **by** $(rule\ contrapos\text{-}nn)$
**then have** $G \in S1$
**by** $(rule\ notnotD)$
**then have** $insert\ G\ S1 = S1$
**by** $(rule\ insert\text{-}absorb)$
**show** *?thesis*
**using** $\langle S1 \notin C\rangle$ **by** $(simp\ only\text{: } simp\text{-}thms(8)\ \langle insert\ G\ S1 = S1\rangle)$
**qed**
**let** $?S1{=}S1 - \{G\}$
**have** $insert\ G\ S = \{G\} \cup S$
**by** $(rule\ insert\text{-}is\text{-}Un)$
**have** $S1 \subseteq \{G\} \cup S$
**using** $\langle S1 \subseteq insert\ G\ S\rangle$ **by** $(simp\ only\text{: } \langle insert\ G\ S = \{G\} \cup S\rangle)$
**have** $1$:$?S1 \subseteq S$
**using** $\langle S1 \subseteq \{G\} \cup S\rangle$ **by** $(simp\ only\text{: } Diff\text{-}subset\text{-}conv)$
**have** $2$:$finite\ ?S1$
**using** $\langle finite\ S1\rangle$ **by** $(simp\ only\text{: } finite\text{-}Diff)$
**have** $insert\ G\ ?S1 = insert\ G\ S1$
**by** $(simp\ only\text{: } insert\text{-}Diff\text{-}single)$
**then have** $3$:$insert\ G\ ?S1 \notin C$
**using** $\langle insert\ G\ S1 \notin C\rangle$ **by** $(simp\ only\text{: } simp\text{-}thms(6{,}8)\ \langle insert\ G\ ?S1 = insert\ G\ S1\rangle)$
**have** $insert\ H\ S \notin (extF\ C)$
**using** *Conj* **by** $(rule\ conjunct2)$
**then have** $\neg(\forall\, S' \subseteq (insert\ H\ S).\ finite\ S' \longrightarrow S' \in C)$
**unfolding** *extF* **by** $(simp\ add\text{: } mem\text{-}Collect\text{-}eq)$
**then have** $Ex2$:$\exists\, S' \subseteq (insert\ H\ S).\ \neg(finite\ S' \longrightarrow S' \in C)$

```
 by (rule sall-simps-not-all)
obtain S2 where S2 ⊆ insert H S and ¬(finite S2 ⟶ S2 ∈ C)
 using Ex2 by (rule subexE)
have finite S2 ∧ S2 ∉ C
 using ‹¬(finite S2 ⟶ S2 ∈ C)› by (simp only: simp-thms(8,25) not-imp)
then have finite S2
 by (rule conjunct1)
have S2 ∉ C
 using ‹finite S2 ∧ S2 ∉ C› by (rule conjunct2)
then have insert H S2 ∉ C
proof –
 have S2 ⊆ S ⟶ finite S2 ⟶ S2 ∈ C
  using E' by (rule allE)
 then have ¬ S2 ⊆ S
  using ‹¬ (finite S2 ⟶ S2 ∈ C)› by (rule mt)
 then have (S2 ⊆ insert H S) ≠ (S2 ⊆ S)
  using ‹S2 ⊆ insert H S› by simp
 then have notSI:¬(S2 ⊆ insert H S ⟷ S2 ⊆ S)
  by blast
 have subsetInsert:H ∉ S2 ⟹ S2 ⊆ insert H S ⟷ S2 ⊆ S
  by (rule subset-insert)
 have ¬(H ∉ S2)
  using notSI subsetInsert by (rule contrapos-nn)
 then have H ∈ S2
  by (rule notnotD)
 then have insert H S2 = S2
  by (rule insert-absorb)
 show ?thesis
  using ‹S2 ∉ C› by (simp only: simp-thms(8) ‹insert H S2 = S2›)
qed
let ?S2=S2 – {H}
have insert H S = {H} ∪ S
 by (rule insert-is-Un)
have S2 ⊆ {H} ∪ S
 using ‹S2 ⊆ insert H S› by (simp only: ‹insert H S = {H} ∪ S›)
have 4:?S2 ⊆ S
 using ‹S2 ⊆ {H} ∪ S› by (simp only: Diff-subset-conv)
have 5:finite ?S2
 using ‹finite S2› by (simp only: finite-Diff)
have insert H ?S2 = insert H S2
```

```
    by (simp only: insert-Diff-single)
   then have 6:insert H ?S2 ∉ C
    using ‹insert H S2 ∉ C› by (simp only: simp-thms(6,8) ‹insert H ?S2 = insert H S2›)
   show False
    using assms(1) assms(2) assms(3) assms(4) assms(5) 1 2 3 4 5 6 by (rule ex3-pcp-SinE-DIS-auxFalse)
  qed
  thus ?thesis
  proof (rule disjE)
   assume insert G S ∈ (extF C)
   have insG:insert G S ∈ (extensionFin C)
    using ‹(extF C) ⊆ (extensionFin C)› ‹insert G S ∈ (extF C)› by (simp only: in-mono)
   have insert G S = {G} ∪ S
    by (rule insert-is-Un)
   then have {G} ∪ S ∈ (extensionFin C)
    using insG ‹insert G S = {G} ∪ S› by (simp only: insG)
   thus ?thesis
    by (rule disjI1)
  next
   assume insert H S ∈ (extF C)
   have insH:insert H S ∈ (extensionFin C)
    using ‹(extF C) ⊆ (extensionFin C)› ‹insert H S ∈ (extF C)› by (simp only: in-mono)
   have insert H S = {H} ∪ S
    by (rule insert-is-Un)
   then have {H} ∪ S ∈ (extensionFin C)
    using insH ‹insert H S = {H} ∪ S› by (simp only: insH)
   thus ?thesis
    by (rule disjI2)
  qed
qed
```

Probados los lemas *ex3-pcp-SinE-CON* y *ex3-pcp-SinE-DIS*, podemos demostrar que $C' = C \cup E$ verifica las condiciones del lema de caracterización de la propiedad de consistencia proposicional para el caso en que $S \in E$, formalizado como *ex3-pcp-SinE*. Dicho lema prueba que, si $C$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, y sea $S \in E$, se verifican las condiciones:

- $\bot \notin S$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in S$ y $\neg\, p \in S$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $S$, se

tiene que $\{\alpha_1,\alpha_2\} \cup S$ pertenece a $C'$.

- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $S$, se tiene que o bien $\{\beta_1\} \cup S$ pertenece a $C'$ o bien $\{\beta_2\} \cup S$ pertenece a $C'$.

**lemma** *ex3-pcp-SinE*:
 **assumes** *pcp C*
  *subset-closed C*
  $S \in (extF\ C)$
 **shows** $\bot \notin S\ \wedge$
  $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)\ \wedge$
  $(\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in (extensionFin\ C))\ \wedge$
  $(\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in (extensionFin\ C) \vee \{H\} \cup S \in (extensionFin\ C))$
**proof** $-$
 **have** *PCP*:$\forall S \in C.$
  $\bot \notin S\ \wedge$
  $(\forall k.\ Atom\ k \in S \longrightarrow \neg\ (Atom\ k) \in S \longrightarrow False)\ \wedge$
  $(\forall F\ G\ H.\ Con\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in C)\ \wedge$
  $(\forall F\ G\ H.\ Dis\ F\ G\ H \longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C)$
  **using** *assms*(1) **by** (*rule pcp-alt1*)
 **have** *E*:$\forall S' \subseteq S.\ finite\ S' \longrightarrow S' \in C$
  **using** *assms*(3) **unfolding** *extF* **by** (*rule CollectD*)
 **have** $\{\} \subseteq S$
  **by** (*rule empty-subsetI*)
 **have** *C1*:$\bot \notin S$
 **proof** (*rule ccontr*)
  **assume** $\neg(\bot \notin S)$
  **then have** $\bot \in S$
   **by** (*rule notnotD*)
  **then have** $\bot \in S \wedge \{\} \subseteq S$
   **using** $\langle\{\} \subseteq S\rangle$ **by** (*rule conjI*)
  **then have** *insert* $\bot\ \{\} \subseteq S$
   **by** (*simp only*: *insert-subset*)
  **have** *finite* $\{\}$
   **by** (*rule finite.emptyI*)
  **then have** *finite* (*insert* $\bot\ \{\}$)
   **by** (*rule finite.insertI*)
  **have** *finite* (*insert* $\bot\ \{\}$) $\longrightarrow$ (*insert* $\bot\ \{\}$) $\in C$
   **using** *E* $\langle$(*insert* $\bot\ \{\}$) $\subseteq S\rangle$ **by** *simp*
  **then have** (*insert* $\bot\ \{\}$) $\in C$

**using** ‹*finite* (*insert* $\bot$ {})› **by** (*rule mp*)
**have** $\bot \notin$ (*insert* $\bot$ {}) $\wedge$
($\forall$ *k*. *Atom k* $\in$ (*insert* $\bot$ {}) $\longrightarrow$ $\neg$ (*Atom k*) $\in$ (*insert* $\bot$ {}) $\longrightarrow$ *False*) $\wedge$
($\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ (*insert* $\bot$ {}) $\longrightarrow$ {*G*, *H*} $\cup$ (*insert* $\bot$ {}) $\in$ *C*) $\wedge$
($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ (*insert* $\bot$ {}) $\longrightarrow$ {*G*} $\cup$ (*insert* $\bot$ {}) $\in$ *C* $\vee$ {*H*} $\cup$ (*insert* $\bot$ {}) $\in$ *C*)
**using** *PCP* ‹(*insert* $\bot$ {}) $\in$ *C*› **by** *blast*
**then have** $\bot \notin$ (*insert* $\bot$ {})
**by** (*rule conjunct1*)
**have** $\bot \in$ (*insert* $\bot$ {})
**by** (*rule insertI1*)
**show** *False*
**using** ‹$\bot \notin$ (*insert* $\bot$ {})› ‹$\bot \in$ (*insert* $\bot$ {})› **by** (*rule notE*)
**qed**
**have** *C2*:$\forall$ *k*. *Atom k* $\in$ *S* $\longrightarrow$ $\neg$ (*Atom k*) $\in$ *S* $\longrightarrow$ *False*
**proof** (*rule allI*)
**fix** *k*
**show** *Atom k* $\in$ *S* $\longrightarrow$ $\neg$(*Atom k*) $\in$ *S* $\longrightarrow$ *False*
**proof** (*rule impI*)+
**assume** *Atom k* $\in$ *S*
**assume** $\neg$(*Atom k*) $\in$ *S*
**let** *?A*=*insert* (*Atom k*) (*insert* ($\neg$(*Atom k*)) {})
**have** *Atom k* $\in$ *?A*
**by** (*simp only*: *insert-iff simp-thms*)
**have** $\neg$(*Atom k*) $\in$ *?A*
**by** (*simp only*: *insert-iff simp-thms*)
**have** *inSubset*:*insert* ($\neg$(*Atom k*)) {} $\subseteq$ *S*
**using** ‹$\neg$(*Atom k*) $\in$ *S*› ‹{} $\subseteq$ *S*› **by** (*simp only*: *insert-subset*)
**have** *?A* $\subseteq$ *S*
**using** *inSubset* ‹*Atom k* $\in$ *S*› **by** (*simp only*: *insert-subset*)
**have** *finite* {}
**by** (*simp only*: *finite.emptyI*)
**then have** *finite* (*insert* ($\neg$(*Atom k*)) {})
**by** (*rule finite.insertI*)
**then have** *finite ?A*
**by** (*rule finite.insertI*)
**have** *finite ?A* $\longrightarrow$ *?A* $\in$ *C*
**using** *E* ‹*?A* $\subseteq$ *S*› **by** (*rule sspec*)
**then have** *?A* $\in$ *C*
**using** ‹*finite ?A*› **by** (*rule mp*)

**have** $\bot \notin ?A$
$\wedge$ ($\forall$ *k*. *Atom k* $\in$ *?A* $\longrightarrow \neg$ (*Atom k*) $\in$ *?A* $\longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *?A* $\longrightarrow$ $\{G,H\} \cup$ *?A* $\in$ *C*)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *?A* $\longrightarrow$ $\{G\} \cup$ *?A* $\in$ *C* $\vee$ $\{H\} \cup$ *?A* $\in$ *C*)
**using** *PCP* ⟨*?A* $\in$ *C*⟩ **by** (*rule bspec*)
**then have** $\forall$ *k*. *Atom k* $\in$ *?A* $\longrightarrow \neg$ (*Atom k*) $\in$ *?A* $\longrightarrow$ *False*
**by** (*iprover elim*: *conjunct2 conjunct1*)
**then have** *Atom k* $\in$ *?A* $\longrightarrow \neg$ (*Atom k*) $\in$ *?A* $\longrightarrow$ *False*
**by** (*iprover elim*: *allE*)
**then have** $\neg$(*Atom k*) $\in$ *?A* $\longrightarrow$ *False*
**using** ⟨*Atom k* $\in$ *?A*⟩ **by** (*rule mp*)
**thus** *False*
**using** ⟨$\neg$(*Atom k*) $\in$ *?A*⟩ **by** (*rule mp*)
**qed**
**qed**
**have** *C3*:$\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G,H\} \cup$ *S* $\in$ (*extensionFin C*)
**proof** (*rule allI*)+
**fix** *F G H*
**show** *Con F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G,H\} \cup$ *S* $\in$ (*extensionFin C*)
**proof** (*rule impI*)+
**assume** *Con F G H*
**assume** *F* $\in$ *S*
**show** $\{G,H\} \cup$ *S* $\in$ (*extensionFin C*)
**using** *assms*(1) *assms*(2) *assms*(3) ⟨*Con F G H*⟩ ⟨*F* $\in$ *S*⟩ **by** (*simp only*: *ex3-pcp-SinE-CON*)
**qed**
**qed**
**have** *C4*:$\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G\} \cup$ *S* $\in$ (*extensionFin C*) $\vee$ $\{H\} \cup$ *S* $\in$ (*extensionFin C*)
**proof** (*rule allI*)+
**fix** *F G H*
**show** *Dis F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G\} \cup$ *S* $\in$ (*extensionFin C*) $\vee$ $\{H\} \cup$ *S* $\in$ (*extensionFin C*)
**proof** (*rule impI*)+
**assume** *Dis F G H*
**assume** *F* $\in$ *S*
**show** $\{G\} \cup$ *S* $\in$ (*extensionFin C*) $\vee$ $\{H\} \cup$ *S* $\in$ (*extensionFin C*)
**using** *assms*(1) *assms*(2) *assms*(3) ⟨*Dis F G H*⟩ ⟨*F* $\in$ *S*⟩ **by** (*rule ex3-pcp-SinE-DIS*)
**qed**
**qed**
**show** *?thesis*

**using** *C1 C2 C3 C4* **by** (*iprover intro*: *conjI*)
**qed**

En conclusión, la prueba detallada completa en Isabelle que demuestra que la extensión $C'$ verifica la propiedad de consistencia proposicional dada una colección $C$ que también la verifique y sea cerrada bajo subconjuntos es la siguiente.

**lemma** *ex3-pcp*:
**assumes** *pcp C*
*subset-closed C*
**shows** *pcp* (*extensionFin C*)
**unfolding** *pcp-alt*
**proof** (*rule ballI*)
**have** *PCP*:$\forall S \in C.$
$\bot \notin S$
$\wedge$ ($\forall k.$ *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall F\ G\ H.$ *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)
$\wedge$ ($\forall F\ G\ H.$ *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in C \vee \{H\} \cup S \in C$)
**using** *assms*(*1*) **by** (*rule pcp-alt1*)
**fix** *S*
**assume** $S \in$ (*extensionFin C*)
**then have** $S \in C \vee S \in$ (*extF C*)
**unfolding** *extensionFin* **by** (*simp only*: *Un-iff*)
**thus** $\bot \notin S \wedge$
($\forall k.$ *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*) $\wedge$
($\forall F\ G\ H.$ *Con F G H* $\longrightarrow F \in S \longrightarrow \{G, H\} \cup S \in$ (*extensionFin C*)) $\wedge$
($\forall F\ G\ H.$ *Dis F G H* $\longrightarrow F \in S \longrightarrow \{G\} \cup S \in$ (*extensionFin C*) $\vee \{H\} \cup S \in$ (*extensionFin C*))
**proof** (*rule disjE*)
**assume** $S \in C$
**show** *?thesis*
**using** *assms* ‹$S \in C$› **by** (*rule ex3-pcp-SinC*)
**next**
**assume** $S \in$ (*extF C*)
**show** *?thesis*
**using** *assms* ‹$S \in$ (*extF C*)› **by** (*rule ex3-pcp-SinE*)
**qed**
**qed**

Por último, podemos dar la prueba completa del lema *1.3.5* en Isabelle.

**lemma** *ex3*:
**assumes** *pcp C*

*subset-closed C*
**shows** $\exists C'.\ C \subseteq C' \wedge pcp\ C' \wedge \textit{finite-character}\ C'$
**proof** $-$
**let** $?C'$=*extensionFin C*
**have** *C1*:$C \subseteq ?C'$
**unfolding** *extensionFin* **by** (*simp only*: *Un-upper1*)
**have** *C2*:*finite-character* ($?C'$)
**using** *assms*(2) **by** (*rule ex3-finite-character*)
**have** *C3*:*pcp* ($?C'$)
**using** *assms* **by** (*rule ex3-pcp*)
**have** $C \subseteq ?C' \wedge pcp\ ?C' \wedge \textit{finite-character}\ ?C'$
**using** *C1 C2 C3* **by** (*iprover intro*: *conjI*)
**thus** *?thesis*
**by** (*rule exI*)
**qed**

# Capítulo 4

# Teorema de existencia de modelo

**Comentario 7**: Añadir introducción.

## 4.1 Sucesiones de conjuntos de una colección

En este apartado daremos una introducción sobre sucesiones de conjuntos de fórmulas a partir de una colección y un conjunto de la misma. De este modo, se mostrarán distintas características sobre las sucesiones y se definirá su límite. En la siguiente sección probaremos que dicho límite constituye un conjunto satisfacible por el lema de Hintikka.

**Comentario 8**: Revisar el párrafo anterior al final

Recordemos que el conjunto de las fórmulas proposicionales se define recursivamente a partir de un alfabeto numerable de variables proposicionales. Por lo tanto, el conjunto de fórmulas proposicionales es igualmente numerable, de modo que es posible enumerar sus elementos. Una vez realizada esta observación, veamos la definición de sucesión de conjuntos de fórmulas proposicionales a partir de una colección y un conjunto de la misma.

**Definición 4.1.1** *Sea $C$ una colección, $S \in C$ y $F_1, F_2, F_3 \dots$ una enumeración de las fórmulas proposicionales. Se define la sucesión de conjuntos de $C$ a partir de $S$ como sigue:*

$$S_0 = S$$

$$S_{n+1} = \begin{cases} S_n \cup \{F_n\} & si \quad S_n \cup \{F_n\} \in C \\ \\ S_n & c.c \end{cases}$$

Para su formalización en Isabelle se ha introducido una instancia en la teoría de *Sintaxis* que indica explícitamente que el conjunto de las fórmulas proposicionales es numerable.

*instance formula* :: (*countable*) *countable by countable-datatype*

De esta manera, se genera paralelamente el método de prueba *countable-datatype* sobre dicho conjunto, que proporciona una enumeración predeterminada de sus elementos junto con herramientas para probar propiedades referentes a la numerabilidad. En particular, en la formalización de la definición *1.4.1* se utilizará la función *from-nat* que, al aplicarla a un número natural $n$, nos devuelve la $n$-ésima fórmula proposicional según una enumeración predeterminada en Isabelle.

**Comentario 9**: Método de prueba

Puesto que la definición de las sucesiones en *1.4.1* se trata de una definición recursiva sobre la estructura recursiva de los números naturales, se formalizará en Isabelle mediante el tipo de funciones primitivas recursivas de la siguiente manera.

**primrec** *pcp-seq* **where**
*pcp-seq C S 0* = *S* |
*pcp-seq C S* (*Suc n*) = (*let Sn* = *pcp-seq C S n*; *Sn1* = *insert* (*from-nat n*) *Sn in*
*if Sn1* ∈ *C then Sn1 else Sn*)

Veamos el primer resultado sobre dichas sucesiones.

**Lema 4.1.2** *Sea $C$ una colección de conjuntos con la propiedad de consistencia proposicional, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de $C$ a partir de $S$ construida según la definición 1.4.1. Entonces, para todo $n \in \mathbb{N}$ se verifica que $S_n \in C$.*

Procedamos con su demostración.

**Demostración:** El resultado se prueba por inducción en los números naturales que conforman los subíndices de la sucesión.

En primer lugar, tenemos que $S_0 = S$ pertenece a $C$ por hipótesis.

Por otro lado, supongamos que $S_n \in C$. Probemos que $S_{n+1} \in C$. Si suponemos que $S_n \cup \{F_n\} \in C$, por definición tenemos que $S_{n+1} = S_n \cup \{F_n\}$, luego pertenece a $C$. En caso contrario, si suponemos que $S_n \cup \{F_n\} \notin C$, por definición tenemos que $S_{n+1} = S_n$, que pertenece igualmente a $C$ por hipótesis de inducción. Por tanto, queda probado el resultado.

□

La formalización y demostración detallada del lema en Isabelle son las siguientes.

**lemma**

```
  assumes pcp C
      S ∈ C
    shows pcp-seq C S n ∈ C
proof (induction n)
  show pcp-seq C S 0 ∈ C
    by (simp only: pcp-seq.simps(1) ‹S ∈ C›)
next
  fix n
  assume HI:pcp-seq C S n ∈ C
  have pcp-seq C S (Suc n) = (let Sn = pcp-seq C S n; Sn1 = insert (from-nat n) Sn in
              if Sn1 ∈ C then Sn1 else Sn)
    by (simp only: pcp-seq.simps(2))
  then have SucDef:pcp-seq C S (Suc n) = (if insert (from-nat n) (pcp-seq C S n) ∈ C then
             insert (from-nat n) (pcp-seq C S n) else pcp-seq C S n)
    by (simp only: Let-def)
  show pcp-seq C S (Suc n) ∈ C
  proof (cases)
    assume 1:insert (from-nat n) (pcp-seq C S n) ∈ C
    have pcp-seq C S (Suc n) = insert (from-nat n) (pcp-seq C S n)
      using SucDef 1 by (simp only: if-True)
    thus pcp-seq C S (Suc n) ∈ C
      by (simp only: 1)
  next
    assume 2:insert (from-nat n) (pcp-seq C S n) ∉ C
    have pcp-seq C S (Suc n) = pcp-seq C S n
      using SucDef 2 by (simp only: if-False)
    thus pcp-seq C S (Suc n) ∈ C
      by (simp only: HI)
  qed
qed
```

Del mismo modo, podemos probar el lema de manera automática en Isabelle.

```
lemma pcp-seq-in: pcp C ⟹ S ∈ C ⟹ pcp-seq C S n ∈ C
proof(induction n)
  case (Suc n)
  hence pcp-seq C S n ∈ C by simp
  thus ?case by (simp add: Let-def)
qed simp
```

Por otro lado, veamos la monotonía de dichas sucesiones.

**Lema 4.1.3** *Toda sucesión de conjuntos construida a partir de una colección y un conjunto según la definición 1.4.1 es monótona.*

En Isabelle, se formaliza de la siguiente forma.

**lemma** *pcp-seq C S n* $\subseteq$ *pcp-seq C S* (*Suc n*)
 **oops**

Procedamos con la demostración del lema.

**Demostración:** Sea una colección de conjuntos $C$, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de $C$ a partir de $S$ según la definición *1.4.1*. Para probar que $\{S_n\}$ es monótona, basta probar que $S_n \subseteq S_{n+1}$ para todo $n \in \mathbb{N}$. En efecto, el resultado es inmediato al considerar dos casos para todo $n \in \mathbb{N}$: $S_n \cup \{F_n\} \in C$ o $S_n \cup \{F_n\} \notin C$. Si suponemos que $S_n \cup \{F_n\} \in C$, por definición tenemos que $S_{n+1} = S_n \cup \{F_n\}$, luego es claro que $S_n \subseteq S_{n+1}$. En caso contrario, si $S_n \cup \{F_n\} \notin C$, por definición se tiene que $S_{n+1} = S_n$, obteniéndose igualmente el resultado por la propiedad reflexiva de la contención de conjuntos.

$\square$

La prueba detallada en Isabelle se muestra a continuación.

**lemma** *pcp-seq C S n* $\subseteq$ *pcp-seq C S* (*Suc n*)
**proof** −
 **have** *pcp-seq C S* (*Suc n*) = (*let Sn* = *pcp-seq C S n*; *Sn1* = *insert* (*from-nat n*) *Sn in*
   *if Sn1* $\in$ *C then Sn1 else Sn*)
  **by** (*simp only*: *pcp-seq.simps*(2))
 **then have** *SucDef*:*pcp-seq C S* (*Suc n*) = (*if insert* (*from-nat n*) (*pcp-seq C S n*) $\in$ *C then*
   *insert* (*from-nat n*) (*pcp-seq C S n*) *else pcp-seq C S n*)
  **by** (*simp only*: *Let-def*)
 **thus** *pcp-seq C S n* $\subseteq$ *pcp-seq C S* (*Suc n*)
 **proof** (*cases*)
  **assume** *1*:*insert* (*from-nat n*) (*pcp-seq C S n*) $\in$ *C*
  **have** *pcp-seq C S* (*Suc n*) = *insert* (*from-nat n*) (*pcp-seq C S n*)
   **using** *SucDef 1* **by** (*simp only*: *if-True*)
  **thus** *pcp-seq C S n* $\subseteq$ *pcp-seq C S* (*Suc n*)
   **by** (*simp only*: *subset-insertI*)
 **next**
  **assume** *2*:*insert* (*from-nat n*) (*pcp-seq C S n*) $\notin$ *C*
  **have** *pcp-seq C S* (*Suc n*) = *pcp-seq C S n*
   **using** *SucDef 2* **by** (*simp only*: *if-False*)
  **thus** *pcp-seq C S n* $\subseteq$ *pcp-seq C S* (*Suc n*)
   **by** (*simp only*: *subset-refl*)
 **qed**

**qed**

Del mismo modo, se puede probar automáticamente en Isabelle/HOL.

**lemma** *pcp-seq-monotonicity*:*pcp-seq C S n* $\subseteq$ *pcp-seq C S* (*Suc n*)
**by** (*smt eq-iff pcp-seq.simps*(2) *subset-insertI*)

Por otra lado, para facilitar posteriores demostraciones en Isabelle/HOL, vamos a formalizar el lema anterior empleando la siguiente definición generalizada de monotonía.

**lemma** *pcp-seq-mono*:
**assumes** $n \leq m$
**shows** *pcp-seq C S n* $\subseteq$ *pcp-seq C S m*
**using** *pcp-seq-monotonicity assms* **by** (*rule lift-Suc-mono-le*)

A continuación daremos un lema que permite caracterizar un elemento de la sucesión en función de los anteriores.

**Lema 4.1.4** *Sea $C$ una colección de conjuntos, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de $C$ a partir de $S$ construida según la definición 1.4.1. Entonces, para todos $n$, $m \in \mathbb{N}$ se verifica* $\bigcup_{n \leq m} S_n = S_m$

**Demostración:** En las condiciones del enunciado, la prueba se realiza por inducción en $m \in \mathbb{N}$.

En primer lugar, consideremos el caso base $m = 0$. El resultado se obtiene directamente:

$\bigcup_{n \leq 0} S_n = \bigcup_{n=0} S_n = S_0 = S_m$

Por otro lado, supongamos por hipótesis de inducción que $\bigcup_{n \leq m} S_n = S_m$. Veamos que se verifica $\bigcup_{n \leq m+1} S_n = S_{m+1}$. Observemos que si $n \leq m + 1$, entonces se tiene que, o bien $n \leq m$, o bien $n = m + 1$. De este modo, aplicando la hipótesis de inducción, deducimos lo siguiente.

$\bigcup_{n \leq m+1} S_n = \bigcup_{n \leq m} S_n \cup \bigcup_{n=m+1} S_n = \bigcup_{n \leq m} S_n \cup S_{m+1} = S_m \cup S_{m+1}$

Por la monotonía de la sucesión, se tiene que $S_m \subseteq S_{m+1}$. Luego, se verifica:

$\bigcup_{n \leq m+1} S_n = S_m \cup S_{m+1} = S_{m+1}$

Lo que prueba el resultado.

□

Procedamos a su formalización y demostración detallada. Para ello, emplearemos la unión generalizada en Isabelle/HOL perteneciente a la teoría Complete-Lattices.thy.

Además, la prueba ha precisado del siguiente lema auxiliar que define la imagen de un conjunto con un único elemento.

**Comentario 10**: No está claro qué se usa de la teoría de retículos. Además, la unión que se usa en el enunciado del lema es la unión generaluzada de conjuntos y parece que no se usa nada de retículos en la prueba automática. Tampoco en la prueba en lenguaje natural.

**Comentario 11**: Adaptación de singleton-conv para la imagen de un conjunto.

**lemma** *imageElem*: $\{f\,n \mid n.\ n = a\} = \{f\,a\}$
**proof** $-$
  **have** *1*:$\{n \mid n.\ n = a\} = \{a\}$
    **by** (*simp only*: *singleton-conv simp-thms*(6,40))
  **have** $\{f\,n \mid n.\ n = a\} = f`\{n \mid n.\ n = a\}$
    **by** (*simp only*: *image-Collect simp-thms*)
  **also have** $\ldots = f`\{a\}$
    **by** (*simp only*: *1*)
  **also have** $\ldots = \{f\,a\} \cup f\,`\,\{\}$
    **by** (*simp add*: *image-insert*)
  **also have** $\ldots = \{f\,a\} \cup \{\}$
    **by** (*simp only*: *image-empty*)
  **also have** $\ldots = \{f\,a\}$
    **by** (*simp only*: *bounded-semilattice-sup-bot-class.sup-bot.right-neutral*)
  **finally show** *?thesis*
    **by** *this*
**qed**

**Comentario 12**: Adaptación de Collectdisjeq para la imagen de un conjunto.

**lemma** *imageUnDisj*:$\{f\,n \mid n.\ P\,n \vee Q\,n\} = \{f\,n \mid n.\ P\,n\} \cup \{f\,n \mid n.\ Q\,n\}$
  **by** *blast*

De este modo, la prueba detallada en Isabelle/HOL es la siguiente.

**lemma** $\bigcup\,\{pcp\text{-}seq\ C\ S\ n \mid n.\ n \leq m\} = pcp\text{-}seq\ C\ S\ m$
**proof** (*induct m*)

  **have** $\bigcup\,\{pcp\text{-}seq\ C\ S\ n \mid n.\ n \leq 0\} = \bigcup\,\{pcp\text{-}seq\ C\ S\ n \mid n.\ n = 0\}$
    **by** (*simp only*: *le-zero-eq*)
  **also have** $\ldots = \bigcup\ \{pcp\text{-}seq\ C\ S\ 0\}$
    **by** (*simp only*: *imageElem*)

**also have** $\ldots = \mathit{pcp\text{-}seq}\ C\ S\ 0$
**by** $(\mathit{simp\ only}{:}\mathit{cSup\text{-}singleton})$
**finally show** $\bigcup \{\mathit{pcp\text{-}seq}\ C\ S\ n|n.\ n \leq 0\} = \mathit{pcp\text{-}seq}\ C\ S\ 0$
**by** *this*

**next**
**fix** $m$
**assume** $\mathit{HI}{:}\bigcup \{\mathit{pcp\text{-}seq}\ C\ S\ n|n.\ n \leq m\} = \mathit{pcp\text{-}seq}\ C\ S\ m$
**have** $m \leq \mathit{Suc}\ m$
**by** $(\mathit{simp\ only}{:}\ \mathit{add\text{-}0\text{-}right})$
**then have** $\mathit{Mon}{:}\mathit{pcp\text{-}seq}\ C\ S\ m \subseteq \mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)$
**by** $(\mathit{rule\ pcp\text{-}seq\text{-}mono})$
**have** $\bigcup \{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq \mathit{Suc}\ m\} = \bigcup \{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n = \mathit{Suc}\ m \vee n \leq m\}$
**using** *le-Suc-eq* **by** *blast*
**also have** $\ldots = \bigcup (\{\mathit{pcp\text{-}seq}\ C\ S\ n\ |\ n.\ n = \mathit{Suc}\ m\} \cup \{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq m\})$ **using** $[[\mathit{simp\text{-}trace}]]$
**by** $(\mathit{simp\ only}{:}\ \mathit{imageUnDisj})$
**also have** $\ldots = \bigcup (\{\mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)\} \cup \{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq m\})$
**by** $(\mathit{simp\ only}{:}\ \mathit{imageElem})$
**also have** $\ldots = \bigcup (\{\mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)\}) \cup \bigcup (\{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq m\})$
**by** $(\mathit{simp\ only}{:}\ \mathit{Union\text{-}Un\text{-}distrib})$
**also have** $\ldots = (\mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)) \cup \bigcup (\{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq m\})$
**by** $(\mathit{simp\ only}{:}\ \mathit{cSup\text{-}singleton})$
**also have** $\ldots = (\mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)) \cup (\mathit{pcp\text{-}seq}\ C\ S\ m)$
**by** $(\mathit{simp\ only}{:}\ \mathit{HI})$
**also have** $\ldots = \mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)$
**using** *Mon* **by** $(\mathit{simp\ only}{:}\ \mathit{Un\text{-}absorb2})$
**finally show** $\bigcup \{\mathit{pcp\text{-}seq}\ C\ S\ n|n.\ n \leq (\mathit{Suc}\ m)\} = \mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m)$
**by** *this*
**qed**

Análogamente, podemos dar una prueba automática del resultado en Isabelle/HOL.

**lemma** $\mathit{pcp\text{-}seq\text{-}UN}{:}\ \bigcup \{\mathit{pcp\text{-}seq}\ C\ S\ n|n.\ n \leq m\} = \mathit{pcp\text{-}seq}\ C\ S\ m$
**proof**$(\mathit{induction}\ m)$
**case** $(\mathit{Suc}\ m)$
**have** $\{f\ n\ |n.\ n \leq \mathit{Suc}\ m\} = \mathit{insert}\ (f\ (\mathit{Suc}\ m))\ \{f\ n\ |n.\ n \leq m\}$
**for** $f$ **using** *le-Suc-eq* **by** *auto*
**hence** $\{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq \mathit{Suc}\ m\} =$
$\mathit{insert}\ (\mathit{pcp\text{-}seq}\ C\ S\ (\mathit{Suc}\ m))\ \{\mathit{pcp\text{-}seq}\ C\ S\ n\ |n.\ n \leq m\}$ **.**

**hence** $\bigcup\{$*pcp-seq C S n* $|n.\ n \leq$ *Suc m*$\} =$
$\bigcup\{$*pcp-seq C S n* $|n.\ n \leq m\} \cup$ *pcp-seq C S* (*Suc m*) **by** *auto*
**thus** *?case* **using** *Suc pcp-seq-mono* **by** *blast*
**qed** *simp*

Finalmente, definamos el límite de las sucesiones presentadas en la definición *1.4.1*.

**Definición 4.1.5** *Sea C una colección, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de C a partir de S según la definición 1.4.1. Se define el límite de la sucesión de conjuntos de C a partir de S como* $\bigcup_{n=0}^{\infty} S_n$

La definición del límite se formaliza utilizando la unión generalizada de Isabelle como sigue.

**definition** *pcp-lim C S* $\equiv \bigcup\{$*pcp-seq C S n*$|n.$ *True*$\}$

Veamos el primer resultado sobre el límite.

**Lema 4.1.6** *Sea C una colección de conjuntos, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de C a partir de S según la definición 1.4.1. Entonces, para todo $n \in \mathbb{N}$, se verifica:*

$S_n \subseteq \bigcup_{n=0}^{\infty} S_n$

**Demostración:** El resultado se obtiene de manera inmediata ya que, para todo $n \in \mathbb{N}$, se verifica que $S_n \in \{S_n\}_{n=0}^{\infty}$. Por tanto, es claro que $S_n \subseteq \bigcup_{n=0}^{\infty} S_n$.

□

Su formalización y demostración detallada en Isabelle se muestran a continuación.

**lemma** *pcp-seq C S n* $\subseteq$ *pcp-lim C S*
**unfolding** *pcp-lim-def*
**proof** $-$
**have** $n \in \{n \mid n.$ *True*$\}$
**by** (*simp only*: *simp-thms*(*21,38*) *Collect-const if-True UNIV-I*)
**then have** *pcp-seq C S n* $\in$ (*pcp-seq C S*)‘$\{n \mid n.$ *True*$\}$
**by** (*simp only*: *imageI*)
**then have** *pcp-seq C S n* $\in \{$*pcp-seq C S n* $\mid n.$ *True*$\}$
**by** (*simp only*: *image-Collect simp-thms*(*40*))
**thus** *pcp-seq C S n* $\subseteq \bigcup\{$*pcp-seq C S n* $\mid n.$ *True*$\}$
**by** (*simp only*: *Union-upper*)
**qed**

Podemos probarlo de manera automática como sigue.

**lemma** *pcp-seq-sub*: *pcp-seq C S n* $\subseteq$ *pcp-lim C S*
 **unfolding** *pcp-lim-def* **by** *blast*

Por otra parte, mostremos el siguiente lema que relaciona la pertenencia de una fórmula proposicional al límite definido en *1.4.5* y su pertenencia a un elemento de la sucesión definida en *1.4.1*.

**Comentario 13**: El párrafo previo no aporta información significativa. He modificado algo la redacción del lema.

**Lema 4.1.7** *Sea C una colección de conjuntos de fórmulas proposicionales, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de C a partir de S según la definición 1.4.1. Si F es una fórmula tal que $F \in \bigcup_{n=0}^{\infty} S_n$. Entonces existe un $k \in \mathbb{N}$ tal que $F \in S_k$.*

**Demostración:** La prueba es inmediata de la definición del límite de la sucesión de conjuntos $\{S_n\}$: si $F$ pertenece a la unión generalizada $\bigcup_{n=0}^{\infty} S_n$, entonces existe algún conjunto $S_k$ tal que $F \in S_k$. Es decir, existe $k \in \mathbb{N}$ tal que $F \in S_k$, como queríamos demostrar.

$\square$

Su prueba detallada en Isabelle/HOL es la siguiente.

**lemma**
 **assumes** $F \in$ *pcp-lim C S*
 **shows** $\exists k.\ F \in$ *pcp-seq C S k*
**proof** $-$
 **have** $F \in \bigcup$ ((*pcp-seq C S*) ' $\{n \mid n.\ True\}$)
  **using** *assms* **by** (*simp only*: *pcp-lim-def image-Collect simp-thms*(40))
 **then have** $\exists k \in \{n.\ True\}.\ F \in$ *pcp-seq C S k*
  **by** (*simp only*: *UN-iff simp-thms*(40))
 **then have** $\exists k \in UNIV.\ F \in$ *pcp-seq C S k*
  **by** (*simp only*: *UNIV-def*)
 **thus** $\exists k.\ F \in$ *pcp-seq C S k*
  **by** (*simp only*: *bex-UNIV*)
**qed**

Mostremos, a continuación, la demostración automática del resultado.

**lemma** *pcp-lim-inserted-at-ex*:
  $S' \in$ *pcp-lim C S* $\Longrightarrow \exists k.\ S' \in$ *pcp-seq C S k*
 **unfolding** *pcp-lim-def* **by** *blast*

Por último, veamos la siguiente propiedad sobre conjuntos finitos contenidos en el límite de las sucesiones definido en *1.4.5*.

**Lema 4.1.8** *Sea $C$ una colección, $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de $C$ a partir de $S$ según la definición 1.4.1. Si $S'$ es un conjunto finito tal que $S' \subseteq \bigcup_{n=0}^{\infty} S_n$, entonces existe un $k \in \mathbb{N}$ tal que $S' \subseteq S_k$.*

**Demostración:** La prueba del resultado se realiza por inducción sobre la estructura recursiva de los conjuntos finitos.

**Comentario 14**: Dos casos: caso base ( $\{\}$) y paso de inducción (S es de la forma $\{F\} \cup S'$, verificando S' la propiedad y hay que probarla para S).
Redactar la demostración de forma más clara.

En primer lugar, consideremos que el conjunto vacío está contenido en el límite de la sucesión de conjuntos de $C$ a partir de $S$. Como $\{\}$ es subconjunto de todo conjunto, en particular lo es de $S = S_0$, probando así el primer caso.

Por otra parte, sea $S'$ un conjunto finito contenido en el límite de la sucesión de conjuntos de $C$ a partir de $S$, de modo que también está contenido en algún $S_{k'}$ para cierto $k' \in \mathbb{N}$. Sea $F$ una fórmula cualquiera no perteneciente a $S'$. Supongamos que $\{F\} \cup S'$ está también contenido en el límite. Probemos que $\{F\} \cup S'$ está contenido en $S_k$ para algún $k \in \mathbb{N}$.

Como hemos supuesto que $\{F\} \cup S'$ está contenido en el límite, entonces se verifica que $F$ pertenece al límite y $S'$ está contenido en él. Por el lema *1.4.7*, como $F$ pertenece al límite, deducimos que existe un $k \in \mathbb{N}$ tal que $F \in S_k$. Por otro lado, como $S'$ está contenido en el límite, por hipótesis de inducción existe algún $k' \in \mathbb{N}$ tal que $S' \subseteq S_{k'}$. El resultado se obtiene considerando el máximo entre $k$ y $k'$, que notaremos por $k''$. En efecto, por la monotonía de la sucesión, se verifica que tanto $S_k$ como $S_{k'}$ están contenidos en $S_{k''}$. De este modo, como $S' \subseteq S_{k'}$, por la transitividad de la contención de conjuntos se tiene que $S' \subseteq S_{k''}$. Además, como $F \in S_k$, se tiene que $F \in S_{k''}$. Por lo tanto, $\{F\} \cup S' \subseteq S_{k''}$, como queríamos demostrar.

□

Procedamos con la demostración detallada en Isabelle.

**lemma**
 **assumes** *finite* $S'$
  $S' \subseteq$ *pcp-lim* $C$ $S$
  **shows** $\exists k.\ S' \subseteq$ *pcp-seq* $C$ $S$ $k$
 **using** *assms*
**proof** (*induction* $S'$ *rule*: *finite-induct*)
 **case** *empty*
 **have** *pcp-seq* $C$ $S$ $0 = S$
  **by** (*simp only*: *pcp-seq.simps*(1))
 **have** $\{\} \subseteq S$

**by** (*rule order-bot-class.bot.extremum*)
**then have** $\{\} \subseteq$ *pcp-seq C S 0*
**by** (*simp only*: ⟨*pcp-seq C S 0* $=$ *S*⟩)
**then show** *?case*
**by** (*rule exI*)
**next**
**case** (*insert F* $S'$)
**then have** *insert F* $S' \subseteq$ *pcp-lim C S*
**by** (*simp only*: *insert.prems*)
**then have** *C*:*F* $\in$ (*pcp-lim C S*) $\wedge$ $S' \subseteq$ *pcp-lim C S*
**by** (*simp only*: *insert-subset*)
**then have** $S' \subseteq$ *pcp-lim C S*
**by** (*rule conjunct2*)
**then have** *EX1*:$\exists$ *k*. $S' \subseteq$ *pcp-seq C S k*
**by** (*simp only*: *insert.IH*)
**obtain** *k1* **where** $S' \subseteq$ *pcp-seq C S k1*
**using** *EX1* **by** (*rule exE*)
**have** *F* $\in$ *pcp-lim C S*
**using** *C* **by** (*rule conjunct1*)
**then have** *EX2*:$\exists$ *k*. *F* $\in$ *pcp-seq C S k*
**by** (*rule pcp-lim-inserted-at-ex*)
**obtain** *k2* **where** *F* $\in$ *pcp-seq C S k2*
**using** *EX2* **by** (*rule exE*)
**have** *k1* $\leq$ *max k1 k2*
**by** (*simp only*: *linorder-class.max.cobounded1*)
**then have** *pcp-seq C S k1* $\subseteq$ *pcp-seq C S* (*max k1 k2*)
**by** (*rule pcp-seq-mono*)
**have** *k2* $\leq$ *max k1 k2*
**by** (*simp only*: *linorder-class.max.cobounded2*)
**then have** *pcp-seq C S k2* $\subseteq$ *pcp-seq C S* (*max k1 k2*)
**by** (*rule pcp-seq-mono*)
**have** $S' \subseteq$ *pcp-seq C S* (*max k1 k2*)
**using** ⟨$S' \subseteq$ *pcp-seq C S k1*⟩ ⟨*pcp-seq C S k1* $\subseteq$ *pcp-seq C S* (*max k1 k2*)⟩ **by** (*rule subset-trans*)
**have** *F* $\in$ *pcp-seq C S* (*max k1 k2*)
**using** ⟨*F* $\in$ *pcp-seq C S k2*⟩ ⟨*pcp-seq C S k2* $\subseteq$ *pcp-seq C S* (*max k1 k2*)⟩ **by** (*rule rev-subsetD*)
**then have** *1*:*insert F* $S' \subseteq$ *pcp-seq C S* (*max k1 k2*)
**using** ⟨$S' \subseteq$ *pcp-seq C S* (*max k1 k2*)⟩ **by** (*simp only*: *insert-subset*)
**thus** *?case*
**by** (*rule exI*)
**qed**

Finalmente, su demostración automática en Isabelle/HOL es la siguiente.

**lemma** *finite-pcp-lim-EX*:
 **assumes** *finite* $S'$
  $S' \subseteq$ *pcp-lim* $C$ $S$
  **shows** $\exists k.\ S' \subseteq$ *pcp-seq* $C$ $S$ $k$
 **using** *assms*
**proof**(*induction* $S'$ *rule*: *finite-induct*)
 **case** (*insert* $F$ $S'$)
 **hence** $\exists k.\ S' \subseteq$ *pcp-seq* $C$ $S$ $k$ **by** *fast*
 **then guess** *k1* **..**
 **moreover obtain** *k2* **where** $F \in$ *pcp-seq* $C$ $S$ *k2*
  **by** (*meson pcp-lim-inserted-at-ex insert.prems insert-subset*)
 **ultimately have** *insert* $F$ $S' \subseteq$ *pcp-seq* $C$ $S$ (*max k1 k2*)
  **by** (*meson pcp-seq-mono dual-order.trans insert-subset max.bounded-iff order-refl subsetCE*)
 **thus** *?case* **by** *blast*
**qed** *simp*

## 4.2 El teorema de existencia de modelo

En esta sección demostraremos finalmente el *teorema de existencia de modelo*, el cual prueba que todo conjunto de fórmulas perteneciente a una colección que verifique la propiedad de consistencia proposicional es satisfacible. Para ello, considerando una colección $C$ cualquiera y $S \in C$, empleando resultados anteriores extenderemos la colección a una colección $C''$ que tenga la propiedad de consistencia proposicional, sea cerrada bajo subconjuntos y sea de carácter finito. De este modo, en esta sección probaremos que el límite de la sucesión formada a partir de una colección que tenga dichas condiciones y un conjunto cualquiera $S$ como se indica en la definición *1.4.1* pertenece a la colección. Es más, demostraremos que dicho límite se trata de un conjunto de *Hintikka* luego, por el *teorema de Hintikka*, es satisfacible. Finalmente, como $S$ está contenido en el límite, quedará demostrada la satisfacibilidad del conjunto $S$ al heredarla por contención.

**Comentario 15**: Habrá que modificar el párrafo anterior al final.

En primer lugar, probemos que si $C$ es una colección que verifica la propiedad de consistencia proposicional, es cerrada bajo subconjuntos y es de carácter finito, entonces el límite de toda sucesión de conjuntos de $C$ según la definición *1.4.1* pertenece a $C$.

**Lema 4.2.1** *Sea $C$ una colección de conjuntos que verifica la propiedad de consistencia proposicional, es cerrada bajo subconjuntos y es de carácter finito. Sea $S \in C$ y $\{S_n\}$ la sucesión de* *conjuntos de $C$ a partir de $S$ según la definición 1.4.1. Entonces, el límite de la sucesión está en $C$.*

**Demostración:** Por definición, como $C$ es de carácter finito, para todo conjunto son equivalentes:

1. El conjunto pertenece a $C$.

2. Todo subconjunto finito suyo pertenece a $C$.

De este modo, para demostrar que el límite de la sucesión $\{S_n\}$ pertenece a $C$, basta probar que todo subconjunto finito suyo está en $C$.

Sea $S'$ un subconjunto finito del límite de la sucesión. Por resultados anteriores,

**Comentario 16**: Especificar el índice del resultado

existe un $k \in \mathbb{N}$ tal que $S' \subseteq S_k$. Por tanto, como $S_k \in C$ para todo $k \in \mathbb{N}$ y $C$ es cerrada bajo subconjuntos, por definición se tiene que $S' \in C$, como queríamos demostrar. $\square$

En Isabelle se formaliza y demuestra detalladamente como sigue.

**lemma**
 **assumes** *pcp C*
  $S \in C$
  *subset-closed C*
  *finite-character C*
 **shows** *pcp-lim C S* $\in C$
**proof** $-$
 **have** $\forall S.\ S \in C \longleftrightarrow (\forall S' \subseteq S.$ *finite* $S' \longrightarrow S' \in C)$
  **using** *assms*(4) **unfolding** *finite-character-def* **by** *this*
 **then have** *FC1*:*pcp-lim C S* $\in C \longleftrightarrow (\forall S' \subseteq ($*pcp-lim C S*$).$ *finite* $S' \longrightarrow S' \in C)$
  **by** (*rule allE*)
 **have** *SC*:$\forall S \in C.\ \forall S' \subseteq S.\ S' \in C$
  **using** *assms*(3) **unfolding** *subset-closed-def* **by** *this*
 **have** *FC2*:$\forall S' \subseteq$ *pcp-lim C S*. *finite* $S' \longrightarrow S' \in C$
 **proof** (*rule sallI*)
  **fix** $S'$ :: *'a formula set*
  **assume** $S' \subseteq$ *pcp-lim C S*
  **show** *finite* $S' \longrightarrow S' \in C$
  **proof** (*rule impI*)
   **assume** *finite* $S'$
   **then have** *EX*:$\exists k.\ S' \subseteq$ *pcp-seq C S k*

**using** ‹$S' \subseteq$ *pcp-lim C S*› **by** (*rule finite-pcp-lim-EX*)
**obtain** *k* **where** $S' \subseteq$ *pcp-seq C S k*
**using** *EX* **by** (*rule exE*)
**have** *pcp-seq C S k* $\in C$
**using** *assms*(1) *assms*(2) **by** (*rule pcp-seq-in*)
**have** $\forall S' \subseteq$ (*pcp-seq C S k*). $S' \in C$
**using** *SC* ‹*pcp-seq C S k* $\in C$› **by** (*rule bspec*)
**thus** $S' \in C$
**using** ‹$S' \subseteq$ *pcp-seq C S k*› **by** (*rule sspec*)
**qed**
**qed**
**show** *pcp-lim C S* $\in C$
**using** *FC1 FC2* **by** (*rule forw-subst*)
**qed**

Por otra parte, podemos dar una prueba automática del resultado.

**lemma** *pcp-lim-in*:
**assumes** *c*: *pcp C*
**and** *el*: $S \in C$
**and** *sc*: *subset-closed C*
**and** *fc*: *finite-character C*
**shows** *pcp-lim C S* $\in C$ (**is** *?cl* $\in C$)
**proof** −
**from** *pcp-seq-in*[*OF c el*, *THEN allI*] **have** $\forall n.$ *pcp-seq C S n* $\in C$ **.**
**hence** $\forall m. \bigcup \{$*pcp-seq C S n*$|n.\ n \leq m\} \in C$ **unfolding** *pcp-seq-UN* **.**
**have** $\forall S' \subseteq$ *?cl. finite* $S' \longrightarrow S' \in C$
**proof** *safe*
**fix** $S'$ :: *'a formula set*
**have** *pcp-seq C S* (*Suc* (*Max* (*to-nat* ‘ $S'$))) $\subseteq$ *pcp-lim C S*
**using** *pcp-seq-sub* **by** *blast*
**assume** ‹*finite* $S'$› ‹$S' \subseteq$ *pcp-lim C S*›
**hence** $\exists k.\ S' \subseteq$ *pcp-seq C S k*
**proof**(*induction* $S'$ *rule*: *finite-induct*)
**case** (*insert x* $S'$)
**hence** $\exists k.\ S' \subseteq$ *pcp-seq C S k* **by** *fast*
**then guess** *k1* **..**
**moreover obtain** *k2* **where** $x \in$ *pcp-seq C S k2*
**by** (*meson pcp-lim-inserted-at-ex insert.prems insert-subset*)
**ultimately have** *insert x* $S' \subseteq$ *pcp-seq C S* (*max k1 k2*)
**by** (*meson pcp-seq-mono dual-order.trans insert-subset max.bounded-iff order-refl subsetCE*)

**thus** *?case* **by** *blast*
**qed** *simp*
**with** *pcp-seq-in*[*OF c el*] *sc*
**show** $S' \in C$ **unfolding** *subset-closed-def* **by** *blast*
**qed**
**thus** $?cl \in C$ **using** *fc* **unfolding** *finite-character-def* **by** *blast*
**qed**

Probemos que, además, el límite de las sucesión definida en *1.4.1* se trata de un elemento maximal de la colección que lo define si esta verifica la propiedad de consistencia proposicional y es cerrada bajo subconjuntos.

**Lema 4.2.2** *Sea C una colección de conjuntos que verifica la propiedad de consistencia proposicional y es cerrada bajo subconjuntos, S un conjunto y $\{S_n\}$ la sucesión de conjuntos de C a partir de S según la definición 1.4.1. Entonces, el límite de la sucesión $\{S_n\}$ es un elemento maximal de C.*

**Demostración:** Por definición de elemento maximal, basta probar que para cualquier conjunto $K \in C$ que contenga al límite de la sucesión se tiene que $K$ y el límite coinciden.

La demostración se realizará por reducción al absurdo. Consideremos un conjunto $K \in C$ que contenga estrictamente al límite de la sucesión $\{S_n\}$. De este modo, existe una fórmula $F$ tal que $F \in K$ y $F$ no está en el límite. Supongamos que $F$ es la $n$-ésima fórmula según la enumeración de la definición *1.4.1* utilizada para construir la sucesión.

Por un lado, hemos probado que todo elemento de la sucesión está contenido en el límite, luego en particular obtenemos que $S_{n+1}$ está contenido en el límite. De este modo, como $F$ no pertenece al límite, es claro que $F \notin S_{n+1}$. Además, $\{F\} \cup S_n \notin C$ ya que, en caso contrario, por la definición *1.4.1* de la sucesión obtendríamos que $S_{n+1} = \{F\} \cup S_n$, lo que contradice que $F \notin S_{n+1}$.

Por otro lado, como $S_n$ también está contenida en el límite que, a su vez, está contenido en $K$, se obtiene por transitividad que $S_n \subseteq K$. Además, como $F \in K$, se verifica que $\{F\} \cup S_n \subseteq K$. Como $C$ es una colección cerrada bajo subconjuntos por hipótesis y $K \in C$, por definición se tiene que $\{F\} \cup S_n \in C$, llegando así a una contradicción con lo demostrado anteriormente.

□

Su formalización y prueba detallada en Isabelle/HOL se muestran a continuación.

**lemma**
**assumes** *pcp C*
*subset-closed C*
$K \in C$

*pcp-lim C S* $\subseteq$ *K*
**shows** *pcp-lim C S* $=$ *K*
**proof** (*rule ccontr*)
**assume** *H*:$\neg$(*pcp-lim C S* $=$ *K*)
**have** *CE*:*pcp-lim C S* $\subseteq$ *K* $\wedge$ *pcp-lim C S* $\neq$ *K*
**using** *assms*(4) *H* **by** (*rule conjI*)
**have** *pcp-lim C S* $\subseteq$ *K* $\wedge$ *pcp-lim C S* $\neq$ *K* $\longleftrightarrow$ *pcp-lim C S* $\subset$ *K*
**by** (*simp only*: *psubset-eq*)
**then have** *pcp-lim C S* $\subset$ *K*
**using** *CE* **by** (*rule iffD1*)
**then have** $\exists$ *F*. *F* $\in$ (*K* $-$ (*pcp-lim C S*))
**by** (*simp only*: *psubset-imp-ex-mem*)
**then have** *E*:$\exists$ *F*. *F* $\in$ *K* $\wedge$ *F* $\notin$ (*pcp-lim C S*)
**by** (*simp only*: *Diff-iff*)
**obtain** *F* **where** *F*:*F* $\in$ *K* $\wedge$ *F* $\notin$ *pcp-lim C S*
**using** *E* **by** (*rule exE*)
**have** *F* $\in$ *K*
**using** *F* **by** (*rule conjunct1*)
**have** *F* $\notin$ *pcp-lim C S*
**using** *F* **by** (*rule conjunct2*)
**have** *pcp-seq C S* (*Suc* (*to-nat F*)) $\subseteq$ *pcp-lim C S*
**by** (*rule pcp-seq-sub*)
**then have** *F* $\in$ *pcp-seq C S* (*Suc* (*to-nat F*)) $\longrightarrow$ *F* $\in$ *pcp-lim C S*
**by** (*rule in-mono*)
**then have** *1*:*F* $\notin$ *pcp-seq C S* (*Suc* (*to-nat F*))
**using** ‹*F* $\notin$ *pcp-lim C S*› **by** (*rule mt*)
**have** *2*: *insert F* (*pcp-seq C S* (*to-nat F*)) $\notin$ *C*
**proof** (*rule ccontr*)
**assume** $\neg$(*insert F* (*pcp-seq C S* (*to-nat F*)) $\notin$ *C*)
**then have** *insert F* (*pcp-seq C S* (*to-nat F*)) $\in$ *C*
**by** (*rule notnotD*)
**then have** *C*:*insert* (*from-nat* (*to-nat F*)) (*pcp-seq C S* (*to-nat F*)) $\in$ *C*
**by** (*simp only*: *from-nat-to-nat*)
**have** *pcp-seq C S* (*Suc* (*to-nat F*)) $=$ (*let Sn* $=$ *pcp-seq C S* (*to-nat F*);
*Sn1* $=$ *insert* (*from-nat* (*to-nat F*)) *Sn in if Sn1* $\in$ *C then Sn1 else Sn*)
**by** (*simp only*: *pcp-seq.simps*(2))
**then have** *SucDef*:*pcp-seq C S* (*Suc* (*to-nat F*)) $=$ (*if insert* (*from-nat* (*to-nat F*)) (*pcp-seq C S* (*to-nat F*)) $\in$ *C*
*then insert* (*from-nat* (*to-nat F*)) (*pcp-seq C S* (*to-nat F*)) *else pcp-seq C S* (*to-nat F*))
**by** (*simp only*: *Let-def*)

**then have** *pcp-seq C S* (*Suc* (*to-nat F*)) = *insert* (*from-nat* (*to-nat F*)) (*pcp-seq C S* (*to-nat F*))
**using** *C* **by** (*simp only*: *if-True*)
**then have** *pcp-seq C S* (*Suc* (*to-nat F*)) = *insert F* (*pcp-seq C S* (*to-nat F*))
**by** (*simp only*: *from-nat-to-nat*)
**then have** *F* ∈ *pcp-seq C S* (*Suc* (*to-nat F*))
**by** (*simp only*: *insertI1*)
**show** *False*
**using** ⟨*F* ∉ *pcp-seq C S* (*Suc* (*to-nat F*))⟩ ⟨*F* ∈ *pcp-seq C S* (*Suc* (*to-nat F*))⟩ **by** (*rule notE*)
**qed**
**have** *pcp-seq C S* (*to-nat F*) ⊆ *pcp-lim C S*
**by** (*rule pcp-seq-sub*)
**then have** *pcp-seq C S* (*to-nat F*) ⊆ *K*
**using** *assms*(4) **by** (*rule subset-trans*)
**then have** *insert F* (*pcp-seq C S* (*to-nat F*)) ⊆ *K*
**using** ⟨*F* ∈ *K*⟩ **by** (*simp only*: *insert-subset*)
**have** ∀ *S* ∈ *C*. ∀ *s*⊆*S*. *s* ∈ *C*
**using** *assms*(2) **by** (*simp only*: *subset-closed-def*)
**then have** ∀ *s* ⊆ *K*. *s* ∈ *C*
**using** *assms*(3) **by** (*rule bspec*)
**then have** 3:*insert F* (*pcp-seq C S* (*to-nat F*)) ∈ *C*
**using** ⟨*insert F* (*pcp-seq C S* (*to-nat F*)) ⊆ *K*⟩ **by** (*rule sspec*)
**show** *False*
**using** *2 3* **by** (*rule notE*)
**qed**

Análogamente a resultados anteriores, veamos su prueba automática.

**lemma** *cl-max*:
**assumes** *c*: *pcp C*
**assumes** *sc*: *subset-closed C*
**assumes** *el*: *K* ∈ *C*
**assumes** *su*: *pcp-lim C S* ⊆ *K*
**shows** *pcp-lim C S* = *K* (**is** *?e*)
**proof** (*rule ccontr*)
**assume** ⟨¬*?e*⟩
**with** *su* **have** *pcp-lim C S* ⊂ *K* **by** *simp*
**then obtain** *F* **where** *e*: *F* ∈ *K* **and** *ne*: *F* ∉ *pcp-lim C S* **by** *blast*
**from** *ne* **have** *F* ∉ *pcp-seq C S* (*Suc* (*to-nat F*)) **using** *pcp-seq-sub* **by** *fast*
**hence** *1*: *insert F* (*pcp-seq C S* (*to-nat F*)) ∉ *C* **by** (*simp add*: *Let-def split*: *if-splits*)
**have** *insert F* (*pcp-seq C S* (*to-nat F*)) ⊆ *K* **using** *pcp-seq-sub e su* **by** *blast*
**hence** *insert F* (*pcp-seq C S* (*to-nat F*)) ∈ *C* **using** *sc*

**unfolding** *subset-closed-def* **using** *el* **by** *blast*
**with** *1* **show** *False* ..
**qed**

A continuación mostremos un resultado sobre el límite de la sucesión de *1.4.1* que es consecuencia de que dicho límite sea un elemento maximal de la colección que lo define si esta verifica la propiedad de consistencia proposicional y es cerrada bajo subconjuntos.

**Corolario 4.2.3** *Sea $C$ una colección de conjuntos que verifica la propiedad de consistencia proposicional y es cerrada bajo subconjuntos, $S$ un conjunto, $\{S_n\}$ la sucesión de conjuntos de $C$ a partir de $S$ según la definición 1.4.1 y las fórmulas proposicionales $F$ y $G$. Entonces, si $\{F\} \cup \bigcup_{n=0}^{\infty} S_n \in C$, se verifica que $F \in \bigcup_{n=0}^{\infty} S_n$. De hecho, si $\{F, G\} \cup \bigcup_{n=0}^{\infty} S_n \in C$, se tiene que $F \in \bigcup_{n=0}^{\infty} S_n$ y $G \in \bigcup_{n=0}^{\infty} S_n$.*

**Comentario 17**: No entiendo bien el enunciado del corolario

**Demostración:** Como $C$ es una colección que verifica la propiedad de consistencia proposicional y es cerrada bajo subconjuntos, se tiene que el límite $\bigcup_{n=0}^{\infty} S_n$ es maximal en $C$. Por lo tanto, si suponemos que $\{F\} \cup \bigcup_{n=0}^{\infty} S_n \in C$, como el límite está contenido en dicho conjunto, se cumple que $\{F\} \cup \bigcup_{n=0}^{\infty} S_n = \bigcup_{n=0}^{\infty} S_n$, luego $F$ pertenece al límite, como queríamos demostrar.

En efecto, si suponemos que $\{F, G\} \cup \bigcup_{n=0}^{\infty} S_n \in C$, como hemos visto que el límite es maximal en $C$ y está contenido en $\{F, G\} \cup \bigcup_{n=0}^{\infty} S_n$, se tiene que coincide con dicho conjunto. Por tanto, es claro que tanto $F$ como $G$ pertenecen al límite.

□

Veamos su formalización y prueba detallada en Isabelle/HOL.

**lemma**
**assumes** *pcp C*
**assumes** *subset-closed C*
**shows** *insert F (pcp-lim C S)* $\in$ *C* $\Longrightarrow$ *F* $\in$ *pcp-lim C S*
*insert F (insert G (pcp-lim C S))* $\in$ *C* $\Longrightarrow$ *F* $\in$ *pcp-lim C S* $\wedge$ *G* $\in$ *pcp-lim C S*
**proof** −
**show** *insert F (pcp-lim C S)* $\in$ *C* $\Longrightarrow$ *F* $\in$ *pcp-lim C S*
**proof** −
**assume** *insert F (pcp-lim C S)* $\in$ *C*
**have** *pcp-lim C S* $\subseteq$ *insert F (pcp-lim C S)*
**by** (*rule subset-insertI*)
**have** *pcp-lim C S* = *insert F (pcp-lim C S)*

**using** *assms*(*1*) *assms*(*2*) ‹*insert F* (*pcp-lim C S*) $\in$ *C*› ‹*pcp-lim C S* $\subseteq$ *insert F* (*pcp-lim C S*)› **by** (*rule cl-max*)
**then have** *insert F* (*pcp-lim C S*) $\subseteq$ *pcp-lim C S*
**by** (*rule equalityD2*)
**then have** *F* $\in$ *pcp-lim C S* $\wedge$ *pcp-lim C S* $\subseteq$ *pcp-lim C S*
**by** (*simp only*: *insert-subset*)
**thus** *F* $\in$ *pcp-lim C S*
**by** (*rule conjunct1*)
**qed**
**next**
**show** *insert F* (*insert G* (*pcp-lim C S*)) $\in$ *C* $\Longrightarrow$ *F* $\in$ *pcp-lim C S* $\wedge$ *G* $\in$ *pcp-lim C S*
**proof** (*rule conjI*)
**assume** *insert F* (*insert G* (*pcp-lim C S* )) $\in$ *C*
**have** *pcp-lim C S* $\subseteq$ *insert G* (*pcp-lim C S*)
**by** (*rule subset-insertI*)
**then have** *pcp-lim C S* $\subseteq$ *insert F* (*insert G* (*pcp-lim C S*))
**by** (*rule subset-insertI2*)
**have** *pcp-lim C S* = *insert F* (*insert G* (*pcp-lim C S*))
**using** *assms*(*1*) *assms*(*2*) ‹*insert F* (*insert G* (*pcp-lim C S*)) $\in$ *C*› ‹*pcp-lim C S* $\subseteq$ *insert F* (*insert G* (*pcp-lim C S*))› **by** (*rule cl-max*)
**then have** *insert F* (*insert G* (*pcp-lim C S*)) $\subseteq$ *pcp-lim C S*
**by** (*rule equalityD2*)
**then have** *1*:*F* $\in$ *pcp-lim C S* $\wedge$ (*insert G* (*pcp-lim C S*)) $\subseteq$ *pcp-lim C S*
**by** (*simp only*: *insert-subset*)
**thus** *F* $\in$ *pcp-lim C S*
**by** (*rule conjunct1*)
**have** *insert G* (*pcp-lim C S*) $\subseteq$ *pcp-lim C S*
**using** *1* **by** (*rule conjunct2*)
**then have** *G* $\in$ *pcp-lim C S* $\wedge$ *pcp-lim C S* $\subseteq$ *pcp-lim C S*
**by** (*simp only*: *insert-subset*)
**thus** *G* $\in$ *pcp-lim C S*
**by** (*rule conjunct1*)
**qed**
**qed**

Mostremos su demostración automática.

**lemma** *cl-max*$'$:
**assumes** *c*: *pcp C*
**assumes** *sc*: *subset-closed C*
**shows** *insert F* (*pcp-lim C S*) $\in$ *C* $\Longrightarrow$ *F* $\in$ *pcp-lim C S*
*insert F* (*insert G* (*pcp-lim C S*)) $\in$ *C* $\Longrightarrow$ *F* $\in$ *pcp-lim C S* $\wedge$ *G* $\in$ *pcp-lim C S*

**using** *cl-max*[*OF assms*] **by** *blast*+

El siguiente resultado prueba que el límite de la sucesión definida en *1.4.1* es un conjunto de Hintikka si la colección que lo define verifica la propiedad de consistencia proposicional, es es cerrada bajo subconjuntos y es de carácter finito. Como consecuencia del *teorema de Hintikka*, se trata en particular de un conjunto satisfacible.

**Lema 4.2.4** *Sea $C$ una colección de conjuntos que verifica la propiedad de consistencia proposicional, es es cerrada bajo subconjuntos y es de carácter finito. Sea $S \in C$ y $\{S_n\}$ la sucesión de conjuntos de $C$ a partir de $S$ según la definición 1.4.1. Entonces, el límite de la sucesión $\{S_n\}$ es un conjunto de Hintikka.*

**Demostración:** Para facilitar la lectura, vamos a notar por $L_{SC}$ al límite de la sucesión $\{S_n\}$ descrita en el enunciado.

Por resultados anteriores, como $C$ verifica la propiedad de consistencia proposicional, es es cerrada bajo subconjuntos y es de carácter finito, se tiene que $L_{SC} \in C$. En particular, por verificar la propiedad de consistencia proposicional, por el lema de caracterización de dicha propiedad mediante notación uniforme, se cumplen las siguientes condiciones para $L_{SC}$:

- $\bot \notin L_{SC}$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in L_{SC}$ y $\neg\, p \in L_{SC}$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ tal que $\alpha$ pertenece a $L_{SC}$, se tiene que $\{\alpha_1,\alpha_2\} \cup L_{SC}$ pertenece a $C$.
- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ tal que $\beta$ pertenece a $L_{SC}$, se tiene que o bien $\{\beta_1\} \cup L_{SC}$ pertenece a $C$ o bien $\{\beta_2\} \cup L_{SC}$ pertenece a $C$.

Veamos que $L_{SC}$ es un conjunto de Hintikka probando que cumple las condiciones del lema de caracterización de los conjuntos de Hintikka mediante notación uniforme, es decir, probaremos que $L_{SC}$ verifica:

- $\bot \notin L_{SC}$.
- Dada $p$ una fórmula atómica cualquiera, no se tiene simultáneamente que $p \in L_{SC}$ y $\neg\, p \in L_{SC}$.
- Para toda fórmula de tipo $\alpha$ con componentes $\alpha_1$ y $\alpha_2$ se verifica que si la fórmula pertenece a $L_{SC}$, entonces $\alpha_1$ y $\alpha_2$ también.

- Para toda fórmula de tipo $\beta$ con componentes $\beta_1$ y $\beta_2$ se verifica que si la fórmula pertenece a $L_{SC}$, entonces o bien $\beta_1$ pertenece a $L_{SC}$ o bien $\beta_2$ pertenece a $L_{SC}$.

Observemos que las dos primeras condiciones coinciden con las obtenidas anteriormente para $L_{SC}$ por el lema de caracterización de la propiedad de consistencia proposicional mediante notación uniforme. Veamos que, en efecto, se cumplen el resto de condiciones.

En primer lugar, probemos que para una fórmula $F$ de tipo $\alpha$ y componentes $\alpha_1$ y $\alpha_2$ tal que $F \in L_{SC}$ se verifica que tanto $\alpha_1$ como $\alpha_2$ pertenecen a $L_{SC}$. Por la tercera condición obtenida anteriormente para $L_{SC}$ por el lema de caracterización de la propiedad de consistencia proposicional mediante notación uniforme, se cumple que $\{\alpha_1,\alpha_2\} \cup L_{SC} \in C$. De este modo, como $C$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, por el corolario *1.5.3* se tiene que $\alpha_1 \in L_{SC}$ y $\alpha_2 \in L_{SC}$, como queríamos demostrar.

Por último, demostremos que para una fórmula $F$ de tipo $\beta$ y componentes $\beta_1$ y $\beta_2$ tal que $F \in L_{SC}$ se verifica que o bien $\beta_1 \in L_{SC}$ o bien $\beta_2 \in L_{SC}$. Por la cuarta condición obtenida anteriormente para $L_{SC}$ por el lema de caracterización de la propiedad de consistencia proposicional mediante notación uniforme, se cumple que o bien $\{\beta_1\} \cup L_{SC} \in C$ o bien $\{\beta_2\} \cup L_{SC} \in C$. De este modo, si suponemos que $\{\beta_1\} \cup L_{SC} \in C$, como $C$ tiene la propiedad de consistencia proposicional y es cerrada bajo subconjuntos, por el corolario *1.5.3* se tiene que $\beta_1 \in L_{SC}$. Por tanto, se cumple que o bien $\beta_1 \in L_{SC}$ o bien $\beta_2 \in L_{SC}$. Si suponemos que $\{\beta_2\} \cup L_{SC} \in C$, se observa fácilmente que llegamos a la misma conclusión de manera análoga. Por lo tanto, queda probado el resultado.

□

Veamos su formalización y prueba detallada en Isabelle.

**lemma**
 **assumes** *pcp C*
 **assumes** *subset-closed C*
 **assumes** *finite-character C*
 **assumes** $S \in C$
 **shows** *Hintikka* (*pcp-lim C S*)
**proof** (*rule Hintikka-alt2*)
 **let** *?cl* = *pcp-lim C S*
 **have** *?cl* $\in C$
  **using** *assms*(*1*) *assms*(*4*) *assms*(*2*) *assms*(*3*) **by** (*rule pcp-lim-in*)
 **have** ($\forall S \in C$.
 $\bot \notin S$
$\wedge$ ($\forall k$. *Atom k* $\in S \longrightarrow \neg$ (*Atom k*) $\in S \longrightarrow$ *False*)
$\wedge$ ($\forall F\ G\ H$. *Con F G H* $\longrightarrow F \in S \longrightarrow \{G,H\} \cup S \in C$)

$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *S* $\longrightarrow$ $\{G\} \cup S \in C \vee \{H\} \cup S \in C$))
  **using** *assms*(1) **by** (*rule pcp-alt1*)
 **then have** *d*:$\bot \notin$ *?cl*
$\wedge$ ($\forall$ *k*. *Atom k* $\in$ *?cl* $\longrightarrow$ $\neg$ (*Atom k*) $\in$ *?cl* $\longrightarrow$ *False*)
$\wedge$ ($\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ $\{G,H\} \cup$ *?cl* $\in C$)
$\wedge$ ($\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ $\{G\} \cup$ *?cl* $\in C \vee \{H\} \cup$ *?cl* $\in C$)
  **using** ‹*?cl* $\in$ *C*› **by** (*rule bspec*)
 **then have** *H1*:$\bot \notin$ *?cl*
  **by** (*rule conjunct1*)
 **have** *H2*:$\forall$ *k*. *Atom k* $\in$ *?cl* $\longrightarrow$ $\neg$ (*Atom k*) $\in$ *?cl* $\longrightarrow$ *False*
  **using** *d* **by** (*iprover elim*: *conjunct2 conjunct1*)
 **have** *Con*:$\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ $\{G,H\} \cup$ *?cl* $\in C$
  **using** *d* **by** (*iprover elim*: *conjunct2 conjunct1*)
 **have** *H3*:$\forall$ *F G H*. *Con F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ *G* $\in$ *?cl* $\wedge$ *H* $\in$ *?cl*
 **proof** (*rule allI*)+
  **fix** *F G H*
  **show** *Con F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ *G* $\in$ *?cl* $\wedge$ *H* $\in$ *?cl*
  **proof** (*rule impI*)+
   **assume** *Con F G H*
   **assume** *F* $\in$ *?cl*
   **have** *Con F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ $\{G,H\} \cup$ *?cl* $\in C$
    **using** *Con* **by** (*iprover elim*: *allE*)
   **then have** *F* $\in$ *?cl* $\longrightarrow$ $\{G,H\} \cup$ *?cl* $\in C$
    **using** ‹*Con F G H*› **by** (*rule mp*)
   **then have** $\{G,H\} \cup$ *?cl* $\in C$
    **using** ‹*F* $\in$ *?cl*› **by** (*rule mp*)
   **have** (*insert G* (*insert H ?cl*)) = $\{G,H\} \cup$ *?cl*
    **by** (*rule insertSetElem*)
   **then have** (*insert G* (*insert H ?cl*)) $\in C$
    **using** ‹$\{G,H\} \cup$ *?cl* $\in C$› **by** (*simp only*: ‹(*insert G* (*insert H ?cl*)) = $\{G,H\} \cup$ *?cl*›)
   **have** (*insert G* (*insert H ?cl*)) $\in C \Longrightarrow G \in$ *?cl* $\wedge$ *H* $\in$ *?cl*
    **using** *assms*(1) *assms*(2) **by** (*rule cl-max*$'$)
   **thus** *G* $\in$ *?cl* $\wedge$ *H* $\in$ *?cl*
    **by** (*simp only*: ‹*insert G* (*insert H ?cl*) $\in C$›)
  **qed**
 **qed**
 **have** *Dis*:$\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ $\{G\} \cup$ *?cl* $\in C \vee \{H\} \cup$ *?cl* $\in C$
  **using** *d* **by** (*iprover elim*: *conjunct2 conjunct1*)
 **have** *H4*:$\forall$ *F G H*. *Dis F G H* $\longrightarrow$ *F* $\in$ *?cl* $\longrightarrow$ *G* $\in$ *?cl* $\vee$ *H* $\in$ *?cl*
 **proof** (*rule allI*)+

```
   fix F G H
   show Dis F G H ⟶ F ∈ ?cl ⟶ G ∈ ?cl ∨ H ∈ ?cl
   proof (rule impI)+
    assume Dis F G H
    assume F ∈ ?cl
    have Dis F G H ⟶ F ∈ ?cl ⟶ {G} ∪ ?cl ∈ C ∨ {H} ∪ ?cl ∈ C
     using Dis by (iprover elim: allE)
    then have F ∈ ?cl ⟶ {G} ∪ ?cl ∈ C ∨ {H} ∪ ?cl ∈ C
     using ‹Dis F G H› by (rule mp)
    then have {G} ∪ ?cl ∈ C ∨ {H} ∪ ?cl ∈ C
     using ‹F ∈ ?cl› by (rule mp)
    thus G ∈ ?cl ∨ H ∈ ?cl
    proof (rule disjE)
     assume {G} ∪ ?cl ∈ C
     have insert G ?cl = {G} ∪ ?cl
      by (rule insert-is-Un)
     have insert G ?cl ∈ C
      using ‹{G} ∪ ?cl ∈ C› by (simp only: ‹insert G ?cl = {G} ∪ ?cl›)
     have insert G ?cl ∈ C ⟹ G ∈ ?cl
      using assms(1) assms(2) by (rule cl-max')
     then have G ∈ ?cl
      by (simp only: ‹insert G ?cl ∈ C›)
     thus G ∈ ?cl ∨ H ∈ ?cl
      by (rule disjI1)
    next
     assume {H} ∪ ?cl ∈ C
     have insert H ?cl = {H} ∪ ?cl
      by (rule insert-is-Un)
     have insert H ?cl ∈ C
      using ‹{H} ∪ ?cl ∈ C› by (simp only: ‹insert H ?cl = {H} ∪ ?cl›)
     have insert H ?cl ∈ C ⟹ H ∈ ?cl
      using assms(1) assms(2) by (rule cl-max')
     then have H ∈ ?cl
      by (simp only: ‹insert H ?cl ∈ C›)
     thus G ∈ ?cl ∨ H ∈ ?cl
      by (rule disjI2)
    qed
   qed
  qed
  show ⊥ ∉ ?cl ∧
```

```
  (∀ k. Atom k ∈ ?cl ⟶ ¬ (Atom k) ∈ ?cl ⟶ False) ∧
  (∀ F G H. Con F G H ⟶ F ∈ ?cl ⟶ G ∈ ?cl ∧ H ∈ ?cl) ∧
  (∀ F G H. Dis F G H ⟶ F ∈ ?cl ⟶ G ∈ ?cl ∨ H ∈ ?cl)
  using H1 H2 H3 H4 by (iprover intro: conjI)
qed
```

Del mismo modo, podemos probar el resultado de manera automática como sigue.

```
lemma pcp-lim-Hintikka:
 assumes c: pcp C
 assumes sc: subset-closed C
 assumes fc: finite-character C
 assumes el: S ∈ C
 shows Hintikka (pcp-lim C S)
proof −
 let ?cl = pcp-lim C S
 have ?cl ∈ C using pcp-lim-in[OF c el sc fc] .
 from c[unfolded pcp-alt, THEN bspec, OF this]
 have d: ⊥ ∉ ?cl
  Atom k ∈ ?cl ⟹ ¬ (Atom k) ∈ ?cl ⟹ False
  Con F G H ⟹ F ∈ ?cl ⟹ insert G (insert H ?cl) ∈ C
  Dis F G H ⟹ F ∈ ?cl ⟹ insert G ?cl ∈ C ∨ insert H ?cl ∈ C
 for k F G H by force+
 have Con F G H ⟹ F ∈ ?cl ⟹ G ∈ ?cl ∧ H ∈ ?cl
    Dis F G H ⟹ F ∈ ?cl ⟹ G ∈ ?cl ∨ H ∈ ?cl
  for F G H
    by(auto dest: d(3−) cl-max'[OF c sc])
 with d(1,2) show ?thesis unfolding Hintikka-alt by fast
qed
```

Finalmente, vamos a demostrar el *teorema de existencia de modelo*. Para ello precisaremos de un resultado que indica que la satisfacibilidad de conjuntos de fórmulas se hereda por la contención.

**Lema 4.2.5** *Todo subconjunto de un conjunto de fórmulas satisfacible es satisfacible.*

**Demostración:** Sea $B$ un conjunto de fórmulas satisfacible y $A \subseteq B$. Veamos que $A$ es satisfacible. Por definición, como $B$ es satisfacible, existe una interpretación $\mathcal{A}$ que es modelo de cada fórmula de $B$. Como $A \subseteq B$, en particular $\mathcal{A}$ es modelo de toda fórmula de $A$. Por tanto, $A$ es satisfacible, ya que existe una interpretación que es modelo de todas sus fórmulas.

□

Su prueba detallada en Isabelle/HOL es la siguiente.

**lemma** *sat-mono*:
 **assumes** $A \subseteq B$
 *sat B*
 **shows** *sat A*
 **unfolding** *sat-def*
**proof** $-$
 **have** *satB*:$\exists \mathcal{A}. \forall F \in B. \mathcal{A} \models F$
  **using** *assms*(2) **by** (*simp only*: *sat-def*)
 **obtain** $\mathcal{A}$ **where** $\forall F \in B. \mathcal{A} \models F$
  **using** *satB* **by** (*rule exE*)
 **have** $\forall F \in A. \mathcal{A} \models F$
 **proof** (*rule ballI*)
  **fix** $F$
  **assume** $F \in A$
  **have** $F \in A \longrightarrow F \in B$
   **using** *assms*(1) **by** (*rule in-mono*)
  **then have** $F \in B$
   **using** $\langle F \in A \rangle$ **by** (*rule mp*)
  **show** $\mathcal{A} \models F$
   **using** $\langle \forall F \in B. \mathcal{A} \models F \rangle$ $\langle F \in B \rangle$ **by** (*rule bspec*)
 **qed**
 **thus** $\exists \mathcal{A}. \forall F \in A. \mathcal{A} \models F$
  **by** (*simp only*: *exI*)
**qed**

De este modo, procedamos finalmente con la demostración del teorema.

**Teorema 4.2.6 (Teorema de Existencia de Modelo)** *Todo conjunto de fórmulas perteneciente a una colección que verifique la propiedad de consistencia proposicional es satisfacible.*

**Demostración:** Sea $C$ una colección de conjuntos de fórmulas proposicionales que verifique la propiedad de consistencia proposicional y $S \in C$. Vamos a probar que $S$ es satisfacible.

En primer lugar, como $C$ verifica la propiedad de consistencia proposicional, por el lema *1.3.3* podemos extenderla a una colección $C'$ que también verifique la propiedad y sea cerrada bajo subconjuntos. A su vez, por el lema *1.3.5*, como la extensión $C'$ es una colección con la propiedad de consistencia proposicional y cerrada bajo subconjuntos, podemos extenderla a otra colección $C''$ que también verifica la propiedad de consistencia proposicional y sea de carácter finito. De este modo, por la transitividad de la contención, es claro que $C''$ es una extensión de $C$, luego $S \in C''$ por hipótesis. Por otro lado, por el lema *1.3.4*, como $C''$ es de carácter finito, se tiene que es cerrada bajo subconjuntos.

En suma, $C''$ es una extensión de $C$ que verifica la propiedad de consistencia proposicional, es cerrada bajo subconjuntos y es de carácter finito. Luego, por el lema *1.5.4*, el límite de la sucesión $\{S_n\}$ de conjuntos de $C''$ a partir de $S$ según la definición *1.4.1* es un conjunto de Hintikka. Por tanto, por el *teorema de Hintikka*, se trata de un conjunto satisfacible.

Finalmente, puesto que para todo $n \in \mathbb{N}$ se tiene que $S_n$ está contenido en el límite, en particular el conjunto $S_0$ está contenido en él. Por definición de la sucesión, dicho conjunto coincide con $S$. Por tanto, como $S$ está contenido en el límite que es un conjunto satisfacible, queda demostrada la satisfacibilidad de $S$.

$\square$

**Comentario 18**: Tal vez sería buena idea hacer un grafo similar al de ex3.

Mostremos su formalización y demostración detallada en Isabelle.

**theorem**
 **fixes** *S* :: *'a* :: *countable formula set*
 **assumes** *pcp C*
 **assumes** *S* ∈ *C*
 **shows** *sat S*
**proof** −
 **have** *pcp C* ⟹ ∃ *C'*. *C* ⊆ *C'* ∧ *pcp C'* ∧ *subset-closed C'*
  **by** (*rule ex1*)
 **then have** *E1*:∃ *C'*. *C* ⊆ *C'* ∧ *pcp C'* ∧ *subset-closed C'*
  **by** (*simp only*: *assms*(*1*))
 **obtain** *Ce'* **where** *H1*:*C* ⊆ *Ce'* ∧ *pcp Ce'* ∧ *subset-closed Ce'*
  **using** *E1* **by** (*rule exE*)
 **have** *C* ⊆ *Ce'*
  **using** *H1* **by** (*rule conjunct1*)
 **have** *pcp Ce'*
  **using** *H1* **by** (*iprover elim*: *conjunct2 conjunct1*)
 **have** *subset-closed Ce'*
  **using** *H1* **by** (*iprover elim*: *conjunct2 conjunct1*)
 **have** *E2*:∃ *Ce*. *Ce'* ⊆ *Ce* ∧ *pcp Ce* ∧ *finite-character Ce*
  **using** ‹*pcp Ce'*› ‹*subset-closed Ce'*› **by** (*rule ex3*)
 **obtain** *Ce* **where** *H2*:*Ce'* ⊆ *Ce* ∧ *pcp Ce* ∧ *finite-character Ce*
  **using** *E2* **by** (*rule exE*)
 **have** *Ce'* ⊆ *Ce*

```
  using H2 by (rule conjunct1)
 then have Subset:C ⊆ Ce
  using ‹C ⊆ Ce'› by (simp only: subset-trans)
 have Pcp:pcp Ce
  using H2 by (iprover elim: conjunct2 conjunct1)
 have FC:finite-character Ce
  using H2 by (iprover elim: conjunct2 conjunct1)
 then have SC:subset-closed Ce
  by (rule ex2)
 have S ∈ C ⟶ S ∈ Ce
  using ‹C ⊆ Ce› by (rule in-mono)
 then have S ∈ Ce
  using assms(2) by (rule mp)
 have Hintikka (pcp-lim Ce S)
  using Pcp SC FC ‹S ∈ Ce› by (rule pcp-lim-Hintikka)
 then have sat (pcp-lim Ce S)
  by (rule Hintikkaslemma)
 have pcp-seq Ce S 0 = S
  by (simp only: pcp-seq.simps(1))
 have pcp-seq Ce S 0 ⊆ pcp-lim Ce S
  by (rule pcp-seq-sub)
 then have S ⊆ pcp-lim Ce S
  by (simp only: ‹pcp-seq Ce S 0 = S›)
 thus sat S
  using ‹sat (pcp-lim Ce S)› by (rule sat-mono)
qed
```

Finalmente, demostremos el teorema de manera automática.

```
theorem pcp-sat:
 fixes S :: 'a :: countable formula set
 assumes c: pcp C
 assumes el: S ∈ C
 shows sat S
proof −
 from c obtain Ce where
   C ⊆ Ce pcp Ce subset-closed Ce finite-character Ce
   using ex1[where 'a='a] ex2[where 'a='a] ex3[where 'a='a]
  by (meson dual-order.trans ex2)
 have S ∈ Ce using ‹C ⊆ Ce› el ..
 with pcp-lim-Hintikka ‹pcp Ce› ‹subset-closed Ce› ‹finite-character Ce›
 have  Hintikka (pcp-lim Ce S) .
```

**with** *Hintikkaslemma* **have** *sat* (*pcp-lim Ce S*) **.**
**moreover have** $S \subseteq$ *pcp-lim Ce S*
**using** *pcp-seq.simps*(1) *pcp-seq-sub* **by** *fast*
**ultimately show** *?thesis* **unfolding** *sat-def* **by** *fast*
**qed**

# Apéndice A

# Lemas de HOL usados

En este glosario se recoge la lista de los lemas y reglas usadas indicando la página del libro de HOL donde se encuentran.

## A.1 La base de lógica de primer orden (2)

En Isabelle corresponde a la teoría HOL.thy

### A.1.1 Lógica primitiva (2.1)

#### A.1.1.1 Conectivas y cuantificadores definidos (2.1.2)

- (p.34) $\neg P \equiv P \longrightarrow \mathit{False}$ (*not-def*)

#### A.1.1.2 Axiomas y definiciones básicas (2.1.4)

- (p.36) $\dfrac{\dfrac{P}{Q}}{P \longrightarrow Q}$ (*impI*)
- (p.36) $\dfrac{(P \longrightarrow Q) \land P}{Q}$ (*mp*)

## A.1.2 Reglas fundamentales (2.2)

### A.1.2.1 Reglas de congruencia para aplicaciones (2.2.2)

- (p.37) $\dfrac{x = y}{f\,x = f\,y}$ (*arg-cong*)
- (p.37) $\dfrac{a = b \wedge c = d}{f\,a\,c = f\,b\,d}$ (*arg-cong2*)

### A.1.2.2 Igualdad de booleanos - *iff* (2.2.3)

- (p.38) $\dfrac{Q = P \wedge Q}{P}$ (*iffD1*)

### A.1.2.3 Cuantificador universal I (2.2.5)

- (p.38) $\dfrac{\forall x.\, P\,x \qquad \dfrac{P\,x}{R}}{R}$ (*allE*)

### A.1.2.4 Negación (2.2.7)

- (p.39) $\dfrac{\dfrac{P}{False}}{\neg\, P}$ (*notI*)
- (p.39) $\dfrac{\neg\, P \wedge P}{R}$ (*notE*)

### A.1.2.5 Implicación (2.2.8)

- (p.40) $\dfrac{Q \qquad \dfrac{P}{\neg\, Q}}{\neg\, P}$ (*contrapos-pn*)

#### A.1.2.6 Disyunción I (2.2.9)

- (p.40) $$\frac{P \vee Q \qquad \dfrac{P}{R} \qquad \dfrac{Q}{R}}{R}$$ (*disjE*)

#### A.1.2.7 Derivación de *iffI* (2.2.10)

- (p.40) $$\frac{\dfrac{P}{Q} \qquad \dfrac{Q}{P}}{P = Q}$$ (*iffI*)

#### A.1.2.8 Cuantificador universal II (2.2.12)

- (p.41) $$\frac{\bigwedge x.\ P\ x}{\forall\, x.\ P\ x}$$ (*allI*)

#### A.1.2.9 Cuantificador existencia (2.2.13)

- (p.41) $$\frac{P\ x}{\exists\, x.\ P\ x}$$ (*exI*)

- (p.41) $$\frac{\exists\, x.\ P\ x \qquad \bigwedge x.\ \dfrac{P\ x}{Q}}{Q}$$ (*exE*)

#### A.1.2.10 Conjunción (2.2.14)

- (p.41) $$\frac{P \wedge Q}{P \wedge Q}$$ (*conjI*)

- (p.41) $$\frac{P \wedge Q}{P}$$ (*conjunct1*)

- (p.41) $$\frac{P \wedge Q}{Q}$$ (*conjunct2*)

#### A.1.2.11 Disyunción II (2.2.15)

- (p.42) $\dfrac{P}{P \vee Q}$ (*disjI1*)
- (p.42) $\dfrac{Q}{P \vee Q}$ (*disjI2*)

#### A.1.2.12 Atomización de conectivas de nivel intermedio (2.2.20)

- (p.46) $(x \equiv y) \equiv x = y$ (*atomize-eq*)

### A.1.3 Configuración del paquete (2.3)

#### A.1.3.1 Simplificadores (2.3.4)

- (p.50) $(\neg\, False) = True$ (*not-False-eq-True*)
- (p.53) $(\nexists x.\ P\ x) = (\forall x.\ \neg\ P\ x)$ (*not-ex*)

## A.2 Grupos, también combinados con órdenes (5)

Los siguientes resultados pertenecen a la teoría de grupos Groups.thy.

### A.2.1 Estructuras abstractas

- (p.109) $sup\ bot\ a = a$ (*sup-bot.left-neutral*)

## A.3 Retículos abstractos (6)

Los resultados expuestos a continuación pertenecen a la teoría de retículos Lattices.thy.

- (p.139) $(sup\ b\ c \leq a) = (b \leq a \wedge c \leq a)$ (*sup.bounded-iff*)

# A.4 Teoría de conjuntos para lógica de orden superior (7)

Los siguientes resultados corresponden a la teoría de conjuntos Set.thy.

## A.4.1 Subconjuntos y cuantificadores acotados (7.2)

- (p.163) $\dfrac{\bigwedge x.\ \dfrac{x \in A}{P\ x}}{\forall\, x{\in}A.\ P\ x}$ (*ballI*)
- (p.163) $\dfrac{(\forall\, x{\in}A.\ P\ x) \wedge x \in A}{P\ x}$ (*bspec*)

## A.4.2 Operaciones básicas (7.3)

### A.4.2.1 Subconjuntos (7.3.1)

- (p.165) $\dfrac{c \in A \wedge A \subseteq B}{c \in B}$ (*rev-subsetD*)
- (p.166) $A \subseteq A$ (*subset-refl*)
- (p.166) $\dfrac{A \subseteq B \wedge B \subseteq C}{A \subseteq C}$ (*subset-trans*)

### A.4.2.2 El conjunto vacío (7.3.3)

- (p.167) $\emptyset \subseteq A$ (*empty-subsetI*)
- (p.167) $\mathit{Ball}\ \emptyset\ P = \mathit{True}$ (*ball-empty*)
- (p.167) $\mathit{Bex}\ \emptyset\ P = \mathit{False}$ (*bex-empty*)

#### A.4.2.3 Unión binaria (7.3.8)

- (p.169) $(c \in A \cup B) = (c \in A \vee c \in B)$ (*Un-iff*)
- (p.169) $\dfrac{c \in A}{c \in A \cup B}$ (*UnI1*)
- (p.170) $\dfrac{c \in B}{c \in A \cup B}$ (*UnI2*)

#### A.4.2.4 Aumentar un conjunto - insertar (7.3.10)

- (p.171) $List.insert\ x\ xs = \{x\} \cup xs$ (*set-insert*)

#### A.4.2.5 Conjuntos unitarios, insertar (7.3.11)

- (p.172) $a \in \{a\}$ (*singletonI*)
- (p.172) $\dfrac{b \in \{a\}}{b = a}$ (*singletonD*)
- (p.172) $(b \in \{a\}) = (b = a)$ (*singleton-iff*)

#### A.4.2.6 Imagen de un conjunto por una función (7.3.12)

- (p.173) $f\ `\ A = \{y \mid \exists x{\in}A.\ y = f\ x\}$ (*image-def*)
- (p.173) $f\ `\ (A \cup B) = f\ `\ A \cup f\ `\ B$ (*image-Un*)
- (p.174) $f\ `\ \emptyset = \emptyset$ (*image-empty*)
- (p.174) $f\ `\ (\{a\} \cup B) = \{f\ a\} \cup f\ `\ B$ (*image-insert*)

### A.4.3 Más operaciones y lemas (7.4)

#### A.4.3.1 Reglas derivadas sobre subconjuntos (7.4.2)

- (p.177) $A \subseteq A \cup B$ (*Un-upper1*)
- (p.177) $B \subseteq A \cup B$ (*Un-upper2*)

#### A.4.3.2 Igualdades sobre la union, intersección, inclusion, etc. (7.4.3)

- (p.179) $\{a\} \cup A = \{a\} \cup A$ (*insert-is-Un*)
- (p.181) $A \cup A = A$ (*Un-absorb*)
- (p.181) $A \cup \emptyset = A$ (*Un-empty-right*)
- (p.182) $\{a\} \cup B \cup C = \{a\} \cup (B \cup C)$ (*Un-insert-left*)
- (p.187) $(\forall x{\in}A.\ P\ x \vee Q) = ((\forall x{\in}A.\ P\ x) \vee Q)$
  $(\forall x{\in}A.\ P \vee Q\ x) = (P \vee (\forall x{\in}A.\ Q\ x))$
  $(\forall x{\in}A.\ P \longrightarrow Q\ x) = (P \longrightarrow (\forall x{\in}A.\ Q\ x))$
  $(\forall x{\in}A.\ P\ x \longrightarrow Q) = ((\exists x{\in}A.\ P\ x) \longrightarrow Q)$
  $(\forall x{\in}\emptyset.\ P\ x) = True$
  $(\forall x{\in}UNIV.\ P\ x) = (\forall x.\ P\ x)$
  $(\forall x{\in}\{a\} \cup B.\ P\ x) = (P\ a \wedge (\forall x{\in}B.\ P\ x))$
  $(\forall x{\in}Collect\ Q.\ P\ x) = (\forall x.\ Q\ x \longrightarrow P\ x)$
  $(\forall x{\in}f\ `\ A.\ P\ x) = (\forall x{\in}A.\ P\ (f\ x))$
  $(\neg\ (\forall x{\in}A.\ P\ x)) = (\exists x{\in}A.\ \neg\ P\ x)$ (*ball-simps*)
- (p.187) $(\exists x{\in}A.\ P\ x \wedge Q) = ((\exists x{\in}A.\ P\ x) \wedge Q)$
  $(\exists x{\in}A.\ P \wedge Q\ x) = (P \wedge (\exists x{\in}A.\ Q\ x))$
  $(\exists x{\in}\emptyset.\ P\ x) = False$
  $(\exists x{\in}UNIV.\ P\ x) = (\exists x.\ P\ x)$
  $(\exists x{\in}\{a\} \cup B.\ P\ x) = (P\ a \vee (\exists x{\in}B.\ P\ x))$
  $(\exists x{\in}Collect\ Q.\ P\ x) = (\exists x.\ Q\ x \wedge P\ x)$
  $(\exists x{\in}f\ `\ A.\ P\ x) = (\exists x{\in}A.\ P\ (f\ x))$
  $(\neg\ (\exists x{\in}A.\ P\ x)) = (\forall x{\in}A.\ \neg\ P\ x)$ (*bex-simps*)

#### A.4.3.3 Monotonía de varias operaciones (7.4.4)

- (p.188) $\dfrac{A \subseteq C \wedge B \subseteq D}{A \cup B \subseteq C \cup D}$ (*Un-mono*)
- (p.188) $P \longrightarrow P$ (*imp-refl*)

- (p.188) $\dfrac{Q \longrightarrow P}{\neg P \longrightarrow \neg Q}$ (*not-mono*)

## A.5 Nociones sobre funciones (9)

En Isabelle, la teoría de funciones se corresponde con Fun.thy.

### A.5.1 Actualización de funciones (9.6)

- (p.212) $f(a := b) = (\lambda x.\ \textsf{if}\ x = a\ \textsf{then}\ b\ \textsf{else}\ f\ x)$ (*fun-upd-def*)
- (p.213) $\dfrac{z \neq x}{(f(x := y))\ z = f\ z}$ (*fun-upd-other*)

## A.6 Retículos completos (10)

En Isabelle corresponde a la teoría Complete-Lattices.thy.

### A.6.1 Retículos completos en conjuntos (10.6)

#### A.6.1.1 Unión (10.6.3)

- (p.238) $\bigcup \emptyset = \emptyset$ (*Union-empty*)

## A.7 Conjuntos finitos (18)

A continuación se muestran resultados relativos a la teoría Finite-Set.thy.

### A.7.1 Predicado de conjuntos finitos (18.1)

- (p.419) *finite A* (*finite*)

### A.7.2 Finitud y operaciones de conjuntos comunes (18.2)

- (p.422) $\dfrac{\mathit{finite}\ F \wedge \mathit{finite}\ G}{\mathit{finite}\ (F \cup G)}$ (*finite-UnI*)
- (p.423) $\mathit{finite}\ (\{a\} \cup A) = \mathit{finite}\ A$ (*finite-insert*)

## A.8 Composición de functores naturales acotados (33)

En esta sección se muestran resultados pertenecientes a la teoría de composición de functores naturales acotados de Isabelle BNFComposition.thy.

- (p.718) $\bigcup\ (f\ `\ (\{a\} \cup B)) = f\ a \cup \bigcup\ (f\ `\ B)$ (*Union-image-insert*)

## A.9 El tipo de datos de la listas finitas (66)

En esta sección se muestran resultados sobre listas finitas dentro de la teoría de listas de Isabelle List.thy.

- (p.1169) $[] = \emptyset$
  $x21 \cdot x22 = \{x21\} \cup x22$ (*list.set*)

### A.9.1 Funciones básicas de procesamiento de listas (66.1)

#### A.9.1.1 Función *set*

- (p.1195) $xs\ @\ ys = xs \cup ys$ (*set-append*)

# Bibliografía


[1] José A. Alonso. Temas de "Lógica matemática y fundamentos (2018–19)". Technical report, Univ. de Sevilla, 2019. En https://www.cs.us.es/~jalonso/cursos/lmf-18/temas.php.

[2] Lawrence C. Paulson Computer Laboratory. Old Introduction to Isabelle. Technical report, University of Cambridge, 2019. En https://isabelle.in.tum.de/website-Isabelle2019/dist/Isabelle2019/doc/intro.pdf.

[3] Christian Doczkal and Gert Smolka. Constructive Formalization of Classical Modal Logic. Technical report, 2011. En http://www.cs.ru.nl/~spitters/coqw_files/paper_1.pdf.

[4] M. Fitting. *First-order Logic and Automated Theorem Proving*. Graduate texts in computer science. Springer, 1996.

[5] L.T.F Gamut. *Introducción a la lógica*. Editorial Universitaria de Buenos Aires, 2002.

[6] John Harrison. An overview of automated reasoning. Technical report, 2014. En https://www.cl.cam.ac.uk/~jrh13/slides/lyon-03feb14/slides.pdf.

[7] Angeliki Koutsoukou-Argyraki. Formalising Mathematics -in praxis, 2019. En https://www.researchgate.net/publication/334549483_formalising_mathematis_-in_praxis_a_mathematician's_first_experiences_with_isabellehol_and_the_why_and_how_of_getting_started.

[8] Dr. Kevin P. Lee. A Guide to Writing Mathematics. En http://cc.kangwon.ac.kr/~kimoon/me/me-132/math-writing.pdf.

[9] F. Félix Lara Martín. Temas 3 de "Ciencias de la computación (2018–19)": Funciones recursivas. Technical report, Univ. de Sevilla, 2019. En http://www.cs.us.es/cursos/cc-2018/Tema-03.pdf.

[10] Julius Michaelis and Tobias Nipkow. Propositional Proof Systems. Technical report, 2020. En https://www.isa-afp.org/browser_info/current/AFP/Propositional_Proof_Systems/document.pdf.

[11] Tobias Nipkow. What's in Main, 2019. En https://isabelle.in.tum.de/website-Isabelle2019/dist/Isabelle2019/doc/main.pdf.

[12] Lawrence C. Paulson Tobias Nipkow and Markus Wenzel. *Isabelle/HOL: A proof assistant for Higher–Order Logic*. Lecture Notes in Computer Science, Vol. 2283, Springer–Verlag, 2019. En https://isabelle.in.tum.de/website-Isabelle2019/dist/Isabelle2019/doc/tutorial.pdf.

[13] Floris van Doorn. Propositional Calculus in Coq. Technical report, 2015. En https://arxiv.org/abs/1503.08744.

[14] Makarius Wenzel. The Isabelle/Isar Implementation, 2019. En https://isabelle.in.tum.de/website-Isabelle2019/dist/Isabelle2019/doc/implementation.pdf.

[15] Makarius Wenzel. The Isabelle/Isar Reference Manual, 2019. En https://isabelle.in.tum.de/website-Isabelle2019/dist/Isabelle2019/doc/isar-ref.pdf.