

PPA

Un asistente de demostración para lógica de primer orden con extracción de testigos usando la traducción de Friedman

Manuel Panichelli

Departamento de Computación, FCEyN, UBA

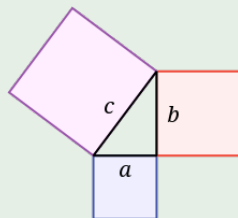
Diciembre 2024

Introducción

- **Teorema:** Afirmación que puede ser *demostrada*.
- **Axiomas:** Afirmaciones que son siempre válidas (sin demostración).
- **Demostración** de un teorema:
 - *Argumento* que establece que el teorema es cierto
 - Usa *reglas de inferencia* a partir de *axiomas* y otros teoremas probados anteriormente.

Ejemplo (Teorema de Pitágoras)

$$a^2 + b^2 = c^2.$$



- **Sistema:** Geometría euclidiana
- **Axioma:** se puede dibujar una línea recta entre dos puntos

- Los **asistentes de demostración** son herramientas que facilitan la escritura y el chequeo de demostraciones por computadora.
- Usos:
 - Formalización de teoremas matemáticos.
 - Verificación de programas.

¹Terrence Tao - Machine Assisted Proof

- Los **asistentes de demostración** son herramientas que facilitan la escritura y el chequeo de demostraciones por computadora.
- Usos:
 - Formalización de teoremas matemáticos.
 - Verificación de programas.
- Ventajas:¹
 - Facilitan la colaboración a gran escala (mediante la confianza en el asistente).

¹Terrence Tao - Machine Assisted Proof

- Los **asistentes de demostración** son herramientas que facilitan la escritura y el chequeo de demostraciones por computadora.
- Usos:
 - Formalización de teoremas matemáticos.
 - Verificación de programas.
- Ventajas:¹
 - Facilitan la colaboración a gran escala (mediante la confianza en el asistente).
 - Habilitan generación automática de demostraciones con IA. Por ej. un *LLM* (como *ChatGPT*) suele devolver alucinaciones, que pueden ser filtradas automáticamente con un asistente.

¹Terrence Tao - Machine Assisted Proof

Constructivos



Coq
(Type theory)

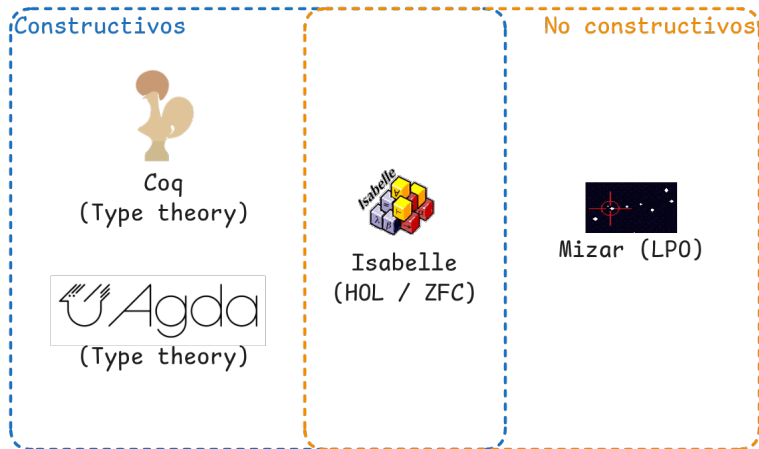


Isabelle
(HOL / ZFC)

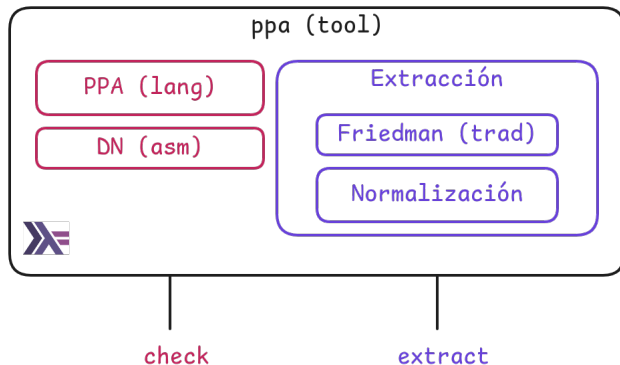
No constructivos



Mizar (LPO)



Extracción de testigos: De una demo de $\exists x.p(x)$, encontrar t tq $p(t)$.
Lógica constructiva = sencillo, no constructiva = complicado.



Diseñamos e implementamos en Haskell **ppa** (*Pani's Proof Assistant*). Dos partes:

- El lenguaje **PPA** para escribir demostraciones.
- Mecanismo de **extracción de testigos** de demostraciones no constructivas (**aporte principal**).

¿Cómo representamos las demostraciones? Ejemplo:

- Tenemos dos axiomas
 - 1 Los alumnos que faltan a los exámenes, los reprueban.
 - 2 Si se reprueba un final, se recursa la materia.
- A partir de ellos, podríamos demostrar que si un alumno falta a un final, entonces recursa la materia.

Representación de demostraciones

¿Cómo representamos las demostraciones? Ejemplo:

- Tenemos dos axiomas
 - 1 Los alumnos que faltan a los exámenes, los reprueban.
 - 2 Si se reprueba un final, se recursa la materia.
- A partir de ellos, podríamos demostrar que si un alumno falta a un final, entonces recursa la materia.

Teorema

Si un alumno falta al final de una materia, entonces la recursa

Demostración.

- Asumo que falta. Quiero ver que recursa.
- Por (1), sabemos que si falta, entonces reprueba. Por lo tanto reprobó.
- Por (2), sabemos que si reprueba, entonces recursa. Por lo tanto recursó.



- **Problema:** Poco precisa. No se puede representar rigurosamente.
- Necesitamos **sistemas deductivos**: sistemas lógicos formales usados para escribir demostraciones
- Usamos **deducción natural**
 - **Lenguaje formal:** lógica de primer orden.
 - **Reglas de inferencia:** Por ejemplo,
 - *modus ponens*: si es cierto $A \rightarrow B$ y A , se puede concluir B
 - *modus tollens*: si es cierto $A \rightarrow B$ y $\neg B$, se puede concluir $\neg A$

Definición (Términos)

Los términos están dados por la gramática:

$$\begin{array}{ll} t ::= x & \text{(variables)} \\ \quad | f(t_1, \dots, t_n) & \text{(funciones)} \end{array}$$

Definición (Fórmulas)

Las fórmulas están dadas por la gramática:

$$\begin{array}{ll} A, B ::= p(t_1, \dots, t_n) & \text{(predicados)} \\ \quad | \perp \mid \top & \text{(falso o } bottom \text{ y verdadero o } top) \\ \quad | A \wedge B \mid A \vee B & \text{(conjunción y disyunción)} \\ \quad | A \rightarrow B \mid \neg A & \text{(implicación y negación)} \\ \quad | \forall x.A \mid \exists x.A & \text{(cuantificador universal y existencial)} \end{array}$$

Deducción natural (DN)

Definiciones

Γ es un **contexto de demostración** y \vdash la **relación de derivabilidad**.

Definición (Reglas de inferencia)

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \text{I} \rightarrow$$

$$\frac{}{\Gamma, A \vdash A} \text{Ax}$$

$$\frac{\Gamma \vdash A \rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B} \text{E} \rightarrow \quad (\textit{modus ponens})$$

Dos tipos para cada conector y cuantificador, dada una fórmula formada con un conector:

- **Introducción:** ¿Cómo la demuestro?
- **Eliminación:** ¿Cómo la uso para demostrar otra?

Ejemplo (Teorema en DN)

Notamos:

- $F \equiv \text{falta}(\text{juan}, \text{final}(\text{logica}))$
- $X \equiv \text{reprueba}(\text{juan}, \text{final}(\text{logica}))$
- $R \equiv \text{recurso}(\text{juan}, \text{logica})$

Afirmamos

$$\left((F \rightarrow X) \wedge (X \rightarrow R) \right) \rightarrow (F \rightarrow R)$$

Ejemplo (Demostración en DN)

$$\vdash \left((F \rightarrow X) \wedge (X \rightarrow R) \right) \rightarrow (F \rightarrow R)$$

Ejemplo (Demostración en DN)

$$\frac{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R}{\vdash ((F \rightarrow X) \wedge (X \rightarrow R)) \rightarrow (F \rightarrow R)} \vdash$$

Ejemplo (Demostración en DN)

$$\frac{\frac{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R}{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R} \text{I} \rightarrow}{\vdash \left((F \rightarrow X) \wedge (X \rightarrow R) \right) \rightarrow (F \rightarrow R)} \text{I} \rightarrow$$

Ejemplo (Demostración en DN)

$$\frac{\frac{\frac{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R}{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R} \text{E} \rightarrow}{\vdash ((F \rightarrow X) \wedge (X \rightarrow R)) \rightarrow (F \rightarrow R)} \text{I} \rightarrow$$

Ejemplo (Demostración en DN)

$$\frac{\frac{\frac{\Gamma \vdash X \rightarrow R}{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R} E \rightarrow}{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R} I \rightarrow}{\vdash \left((F \rightarrow X) \wedge (X \rightarrow R) \right) \rightarrow (F \rightarrow R)} I \rightarrow$$

Ejemplo (Demostración en DN)

$$\begin{array}{c}
 \frac{}{\Gamma \vdash (F \rightarrow X) \wedge (X \rightarrow R)} \text{Ax} \\
 \hline
 \frac{\Gamma \vdash X \rightarrow R}{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R} \text{E}\wedge_2 \\
 \hline
 \frac{\Gamma \vdash X}{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R} \text{E}\rightarrow \\
 \hline
 \frac{}{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R} \text{I}\rightarrow \\
 \hline
 \vdash \left((F \rightarrow X) \wedge (X \rightarrow R) \right) \rightarrow (F \rightarrow R) \text{I}\rightarrow
 \end{array}$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \text{E}\wedge_1$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B} \text{E}\wedge_2$$

Ejemplo (Demostración en DN)

$$\begin{array}{c}
 \frac{\Gamma \vdash (F \rightarrow X) \wedge (X \rightarrow R)}{\Gamma \vdash X \rightarrow R} \text{Ax} \\
 \frac{\Gamma \vdash X \rightarrow R}{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R} \text{E}\wedge_2 \quad \text{□} \\
 \frac{\Gamma = (F \rightarrow X) \wedge (X \rightarrow R), F \vdash R}{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R} \text{E}\rightarrow \\
 \frac{(F \rightarrow X) \wedge (X \rightarrow R) \vdash F \rightarrow R}{\vdash ((F \rightarrow X) \wedge (X \rightarrow R)) \rightarrow (F \rightarrow R)} \text{I}\rightarrow
 \end{array}$$

$$\begin{array}{c}
 \frac{\Gamma \vdash (F \rightarrow X) \wedge (X \rightarrow R)}{\Gamma \vdash F \rightarrow X} \text{Ax} \\
 \frac{\Gamma \vdash F \rightarrow X}{\Gamma \vdash X} \text{E}\wedge_1 \quad \text{□} = \\
 \frac{\Gamma \vdash X}{\Gamma \vdash F} \text{E}\rightarrow \quad \text{Ax}
 \end{array}$$

Otras reglas de inferencia

- $E\perp$, IT
- $I\neg$, $E\neg$
- IV_1 , IV_2 , EV
- $I\forall$, $E\forall$
- $I\exists$, $E\exists$
- $I\wedge$
- LEM

Otras reglas de inferencia

- $E\perp, IT$
- $I\neg, E\neg$
- IV_1, IV_2, EV
- $I\forall, E\forall$
- $I\exists, E\exists$
- $I\wedge$
- LEM

Alfa equivalencia

- Podemos usar $\exists x.p(x)$ y $\exists y.p(y)$ de forma intercambiable.
- Son α -equivalentes (renombrando variables ligadas de forma apropiada, son iguales).

Reglas admisibles

- Mencionamos *modus tollens* pero no aparece en las reglas de inferencia.
- Queremos un sistema lógico **minimal**: no agregamos las reglas **admisibles**, derivables a partir de las existentes.
- Se implementan como funciones o *macros*.

Reglas admisibles

- Mencionamos *modus tollens* pero no aparece en las reglas de inferencia.
- Queremos un sistema lógico **minimal**: no agregamos las reglas **admisibles**, derivables a partir de las existentes.
- Se implementan como funciones o *macros*.

Lema (Modus tollens)

$$\frac{\frac{\frac{}{\Gamma \vdash (A \rightarrow B) \wedge \neg B} Ax}{\Gamma \vdash \neg B} E\wedge_2 \quad \frac{\frac{\frac{}{\Gamma \vdash (A \rightarrow B) \wedge \neg B} Ax}{\Gamma \vdash A \rightarrow B} E\wedge_1 \quad \frac{}{\Gamma \vdash A} Ax}{\Gamma \vdash B} E\rightarrow}{\Gamma = (A \rightarrow B) \wedge \neg B, A \vdash \perp} E\neg}{(A \rightarrow B) \wedge \neg B \vdash \neg A} I\vdash}{\vdash (A \rightarrow B \wedge \neg B) \rightarrow \neg A} I\rightarrow$$

Eliminación de universal

$$\frac{\Gamma \vdash \forall x.A}{\Gamma \vdash A\{x := t\}} \text{E}\forall$$

Eliminación de universal

$$\frac{\Gamma \vdash \forall x.A}{\Gamma \vdash A\{x := t\}} \text{E}\forall$$

Definición (Sustitución)

Notamos como $A\{x := t\}$ a la sustitución de todas las ocurrencias libres de la variable x por el término t en la fórmula A .

Eliminación de universal

$$\frac{\Gamma \vdash \forall x.A}{\Gamma \vdash A\{x := t\}} \text{E}\forall$$

Definición (Sustitución)

Notamos como $A\{x := t\}$ a la sustitución de todas las ocurrencias libres de la variable x por el término t en la fórmula A .

Capturas

Evitamos automáticamente la **captura de variables** (renombrando a fórmula α -equivalente tq no ocurra)

$(\forall y.p(\mathbf{x}, y))\{x := y\} \neq \forall y.p(\mathbf{y}, y)$ (capturada)

$(\forall y.p(\mathbf{x}, y))\{x := y\} = \forall \mathbf{z}.p(\mathbf{y}, \mathbf{z})$ (renombrada)

PPA

*Mathematical Vernacular*² = Mizar + Isar (Isabelle)

Forma *natural* de representar demostraciones matemáticas. Ideas:

²De Freek Wiedijk

*Mathematical Vernacular*² = Mizar + Isar (Isabelle)

Forma *natural* de representar demostraciones matemáticas. Ideas:

- **Deducción natural en estilo de *Fitch***. Notación equivalente, demostraciones como listas de fórmulas en lugar de árboles.

²De Freek Wiedijk

*Mathematical Vernacular*² = Mizar + Isar (Isabelle)

Forma *natural* de representar demostraciones matemáticas. Ideas:

- **Deducción natural en estilo de *Fitch***. Notación equivalente, demostraciones como listas de fórmulas en lugar de árboles.
- **Reglas de inferencia *declarativas***: Afirmar

$$A_1, \dots, A_n \vdash A$$

sin tener que demostrarlo a mano (automático).

²De Freek Wiedijk

$$\textit{Mathematical Vernacular}^2 = \textit{Mizar} + \textit{Isar} \text{ (Isabelle)}$$

Forma *natural* de representar demostraciones matemáticas. Ideas:

- **Deducción natural en estilo de *Fitch***. Notación equivalente, demostraciones como listas de fórmulas en lugar de árboles.
- **Reglas de inferencia *declarativas***: Afirmar

$$A_1, \dots, A_n \vdash A$$

sin tener que demostrarlo a mano (automático).

- **Sintaxis similar a un lenguaje de programación** en lugar al lenguaje natural.

²De Freek Wiedijk

Lenguaje PPA, inspirado en el *Mathematical Vernacular*. Demostraciones son listas de **comandos** que reducen sucesivamente la *tesis* (fórmula a demostrar) hasta agotarla.

Ejemplo demostración

```
1  axiom "ax1": forall A . forall E .  
2      falta(A, E) -> reprueba(A, E)  
3  axiom "ax2": forall A . forall M .  
4      reprueba(A, final(M)) -> recursa(A, M)
```

Lenguaje PPA, inspirado en el *Mathematical Vernacular*. Demostraciones son listas de **comandos** que reducen sucesivamente la *tesis* (fórmula a demostrar) hasta agotarla.

Ejemplo demostración

```
1 axiom "ax1": forall A . forall E .  
2   falta(A, E) -> reprueba(A, E)  
3 axiom "ax2": forall A . forall M .  
4   reprueba(A, final(M)) -> recursa(A, M)  
5  
6 theorem "falta_entonces_recura": forall A . forall M .  
7   falta(A, final(M)) -> recursa(A, M)  
8 proof
```

Lenguaje PPA, inspirado en el *Mathematical Vernacular*. Demostraciones son listas de **comandos** que reducen sucesivamente la *tesis* (fórmula a demostrar) hasta agotarla.

Ejemplo demostración

```
1  axiom "ax1": forall A . forall E .
2      falta(A, E) -> reprueba(A, E)
3  axiom "ax2": forall A . forall M .
4      reprueba(A, final(M)) -> recursa(A, M)
5
6  theorem "falta_entonces_recura": forall A . forall M .
7      falta(A, final(M)) -> recursa(A, M)
8  proof
9      let A
10     let M
```

Lenguaje PPA, inspirado en el *Mathematical Vernacular*. Demostraciones son listas de **comandos** que reducen sucesivamente la *tesis* (fórmula a demostrar) hasta agotarla.

Ejemplo demostración

```
1  axiom "ax1": forall A . forall E .
2      falta(A, E) -> reprueba(A, E)
3  axiom "ax2": forall A . forall M .
4      reprueba(A, final(M)) -> recursa(A, M)
5
6  theorem "falta_entonces_recura": forall A . forall M .
7      falta(A, final(M)) -> recursa(A, M)
8  proof
9      let A
10     let M
11     suppose "falta": falta(A, final(M))
```


Lenguaje PPA, inspirado en el *Mathematical Vernacular*. Demostraciones son listas de **comandos** que reducen sucesivamente la *tesis* (fórmula a demostrar) hasta agotarla.

Ejemplo demostración

```

1  axiom "ax1": forall A . forall E .
2      falta(A, E) -> reprueba(A, E)
3  axiom "ax2": forall A . forall M .
4      reprueba(A, final(M)) -> recursa(A, M)
5
6  theorem "falta_entonces_recura": forall A . forall M .
7      falta(A, final(M)) -> recursa(A, M)
8  proof
9      let A
10     let M
11     suppose "falta": falta(A, final(M))
12     have "reprueba": reprueba(A, final(M)) by "ax1", "falta"

```

Lenguaje PPA, inspirado en el *Mathematical Vernacular*. Demostraciones son listas de **comandos** que reducen sucesivamente la *tesis* (fórmula a demostrar) hasta agotarla.

Ejemplo demostración

```

1  axiom "ax1": forall A . forall E .
2      falta(A, E) -> reprueba(A, E)
3  axiom "ax2": forall A . forall M .
4      reprueba(A, final(M)) -> recursa(A, M)
5
6  theorem "falta_entonces_recura": forall A . forall M .
7      falta(A, final(M)) -> recursa(A, M)
8  proof
9      let A
10     let M
11     suppose "falta": falta(A, final(M))
12     have "reprueba": reprueba(A, final(M)) by "ax1", "falta"
13     thus recursa(A, M) by "ax2", "reprueba"
14 end

```

Comandos y reglas de inferencia

| Regla | Comando |
|------------|-----------------|
| LEM | cases |
| Ax | by |
| $I\exists$ | take |
| $E\exists$ | consider |
| $I\forall$ | let |
| $E\forall$ | by |
| $I\vee_1$ | by |
| $I\vee_2$ | by |
| $E\vee$ | cases |

| Regla | Comando |
|----------------|----------------|
| $I\wedge$ | by |
| $E\wedge_1$ | by |
| $E\wedge_2$ | by |
| $I\rightarrow$ | suppose |
| $E\rightarrow$ | by |
| $I\neg$ | suppose |
| $E\neg$ | by |
| IT | by |
| $E\perp$ | by |

Comandos y reglas de inferencia

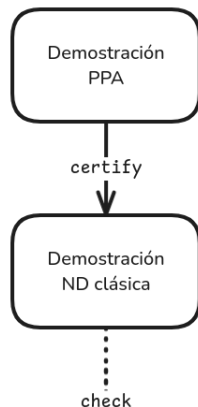
| Regla | Comando |
|------------|-----------------|
| LEM | cases |
| Ax | by |
| $I\exists$ | take |
| $E\exists$ | consider |
| $I\forall$ | let |
| $E\forall$ | by |
| $I\vee_1$ | by |
| $I\vee_2$ | by |
| $E\vee$ | cases |

| Regla | Comando |
|----------------|----------------|
| $I\wedge$ | by |
| $E\wedge_1$ | by |
| $E\wedge_2$ | by |
| $I\rightarrow$ | suppose |
| $E\rightarrow$ | by |
| $I\neg$ | suppose |
| $E\neg$ | by |
| $I\top$ | by |
| $E\perp$ | by |

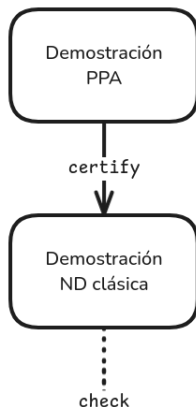
Adicionales:

- **equivalently**: Reduce la tesis a una fórmula equivalente.
- **claim**: Análogo a **have** pero con una sub-demostración.

- Las demostraciones de PPA se **certifican** generando una demostración de deducción natural.
- Evita confiar en la implementación del asistente.



- Las demostraciones de PPA se **certifican** generando una demostración de deducción natural.
- Evita confiar en la implementación del asistente.
- Cumple con el **Criterio de de Bruijn** (sus demostraciones pueden ser chequeadas por un programa independiente)



El certificado de una demostración es recursivo:

```
1  theorem t:  
2    p(v) -> exists X . p(X)  
3  proof  
4    suppose "h": p(v)  
5    take X := v  
6    thus p(v) by "h"  
7  end
```

$$\frac{\frac{\overline{h : p(v) \vdash p(v)} \text{ } \textcolor{red}{Ax_h}}{h : p(v) \vdash \exists x.p(X)} \text{ } \exists}{\vdash p(v) \rightarrow \exists x.p(X)} \text{ } \mapsto_h$$

Figura: Ejemplo de certificado generado para un programa

by - El mecanismo principal de demostración

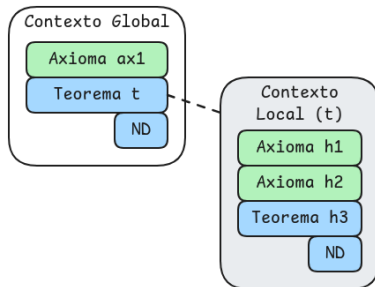
```
thus <form> by <h1>, ..., <hn>  
have <name>: <form> by <h1>, ..., <hn>
```

- Si puede, demuestra **automáticamente** que la fórmula es consecuencia lógica de la justificación.
- Por debajo usa un *solver* completo para lógica proposicional pero *heurístico* para primer orden.

by - El mecanismo principal de demostración

```
thus <form> by <h1>, ..., <hn>  
have <name>: <form> by <h1>, ..., <hn>
```

- Si puede, demuestra **automáticamente** que la fórmula es consecuencia lógica de la justificación.
- Por debajo usa un *solver* completo para lógica proposicional pero *heurístico* para primer orden.
- Toma las hipótesis del **contexto** local o global: fórmulas asumidas o demostradas.



Certificado del by

Teniendo $\Gamma = \{h_1 : B_1, \dots, h_n : B_n\}$, para **thus** A **by** h_1, \dots, h_n :

- 1 Buscamos las hipótesis en el contexto. Queremos demostrar

$$\Gamma \vdash B_1 \wedge \dots \wedge B_n \rightarrow A$$

Certificado del by

Teniendo $\Gamma = \{h_1 : B_1, \dots, h_n : B_n\}$, para **thus** A **by** h_1, \dots, h_n :

- 1 Buscamos las hipótesis en el contexto. Queremos demostrar

$$\Gamma \vdash B_1 \wedge \dots \wedge B_n \rightarrow A$$

- 2 **Razonamos por el absurdo:** Asumiendo la negación buscamos una contradicción

$$\Gamma, \neg(B_1 \wedge \dots \wedge B_n \rightarrow A) \vdash \perp$$

Certificado del by

Teniendo $\Gamma = \{h_1 : B_1, \dots, h_n : B_n\}$, para **thus** A **by** h_1, \dots, h_n :

- 1 Buscamos las hipótesis en el contexto. Queremos demostrar

$$\Gamma \vdash B_1 \wedge \dots \wedge B_n \rightarrow A$$

- 2 **Razonamos por el absurdo:** Asumiendo la negación buscamos una contradicción

$$\Gamma, \neg(B_1 \wedge \dots \wedge B_n \rightarrow A) \vdash \perp$$

- 3 Convertimos la negación a forma normal disyuntiva (**DNF**)

$$\Gamma, (a_1 \wedge \dots \wedge a_n) \vee \dots \vee (b_1 \wedge \dots \wedge b_m) \vdash \perp$$

Certificado del by

Teniendo $\Gamma = \{h_1 : B_1, \dots, h_n : B_n\}$, para **thus** A **by** h_1, \dots, h_n :

- 1 Buscamos las hipótesis en el contexto. Queremos demostrar

$$\Gamma \vdash B_1 \wedge \dots \wedge B_n \rightarrow A$$

- 2 **Razonamos por el absurdo:** Asumiendo la negación buscamos una contradicción

$$\Gamma, \neg(B_1 \wedge \dots \wedge B_n \rightarrow A) \vdash \perp$$

- 3 Convertimos la negación a forma normal disyuntiva (**DNF**)

$$\Gamma, (a_1 \wedge \dots \wedge a_n) \vee \dots \vee (b_1 \wedge \dots \wedge b_m) \vdash \perp$$

- 4 Buscamos una **contradicción** refutando cada cláusula individualmente. Será refutable si

Certificado del by

Teniendo $\Gamma = \{h_1 : B_1, \dots, h_n : B_n\}$, para **thus** A **by** h_1, \dots, h_n :

- 1 Buscamos las hipótesis en el contexto. Queremos demostrar

$$\Gamma \vdash B_1 \wedge \dots \wedge B_n \rightarrow A$$

- 2 **Razonamos por el absurdo:** Asumiendo la negación buscamos una contradicción

$$\Gamma, \neg(B_1 \wedge \dots \wedge B_n \rightarrow A) \vdash \perp$$

- 3 Convertimos la negación a forma normal disyuntiva (**DNF**)

$$\Gamma, (a_1 \wedge \dots \wedge a_n) \vee \dots \vee (b_1 \wedge \dots \wedge b_m) \vdash \perp$$

- 4 Buscamos una **contradicción** refutando cada cláusula individualmente. Será refutable si

- Contiene \perp o dos fórmulas opuestas $(a, \neg a)$,

Teniendo $\Gamma = \{h_1 : B_1, \dots, h_n : B_n\}$, para **thus** A **by** h_1, \dots, h_n :

- 1 Buscamos las hipótesis en el contexto. Queremos demostrar

$$\Gamma \vdash B_1 \wedge \dots \wedge B_n \rightarrow A$$

- 2 **Razonamos por el absurdo:** Asumiendo la negación buscamos una contradicción

$$\Gamma, \neg(B_1 \wedge \dots \wedge B_n \rightarrow A) \vdash \perp$$

- 3 Convertimos la negación a forma normal disyuntiva (**DNF**)

$$\Gamma, (a_1 \wedge \dots \wedge a_n) \vee \dots \vee (b_1 \wedge \dots \wedge b_m) \vdash \perp$$

- 4 Buscamos una **contradicción** refutando cada cláusula individualmente. Será refutable si

- Contiene \perp o dos fórmulas opuestas $(a, \neg a)$,
- Eliminando universales consecutivos y reiniciando el proceso, se consigue una refutación $(\neg p(k), \forall x.p(x))$

Ejemplo sin cuantificadores (1/3)

By sin cuantificadores

```
1 axiom ax1: a -> b
2 axiom ax2: a
3 theorem t: b
4 proof
5   thus b by ax1, ax2
6 end
```


Ejemplo sin cuantificadores (1/3)

By sin cuantificadores

```
1 axiom ax1: a -> b
2 axiom ax2: a
3 theorem t: b
4 proof
5   thus b by ax1, ax2
6 end
```

- ❶ Para certificar **thus** b **by** ax1, ax2 hay que generar una demostración para la implicación

$$\Gamma \vdash ((a \rightarrow b) \wedge a) \rightarrow b$$

Ejemplo sin cuantificadores (1/3)

By sin cuantificadores

```
1 axiom ax1: a -> b
2 axiom ax2: a
3 theorem t: b
4 proof
5   thus b by ax1, ax2
6 end
```

- ❶ Para certificar **thus** b **by** ax1, ax2 hay que generar una demostración para la implicación

$$\Gamma \vdash ((a \rightarrow b) \wedge a) \rightarrow b$$

- ❷ Negamos la fórmula y buscamos una contradicción.

$$\Gamma, \neg[((a \rightarrow b) \wedge a) \rightarrow b] \vdash \perp$$

Razonamiento por el absurdo

Razonamiento por el absurdo en DNF

$$\Gamma \vdash ((a \rightarrow b) \wedge a) \rightarrow b \quad \overset{?}{\rightsquigarrow} \quad \Gamma, \neg[((a \rightarrow b) \wedge a) \rightarrow b] \vdash \perp$$

Razonamiento por el absurdo

Razonamiento por el absurdo en DNF

$$\Gamma \vdash ((a \rightarrow b) \wedge a) \rightarrow b \quad ? \quad \Gamma, \neg[((a \rightarrow b) \wedge a) \rightarrow b] \vdash \perp$$

Teorema (DNeg Elim)

$$\frac{\frac{}{\neg\neg A \vdash A}}{} E_{\neg\neg}$$

$$\frac{}{\Gamma \vdash A \vee \neg A} LEM$$

Teorema (cut)

$$\frac{\Gamma, B \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A} cut$$

Lema (Razonamiento por el absurdo)

$$\frac{\frac{\Gamma, \neg A \vdash \perp}{\Gamma \vdash \neg\neg A} I_{\neg}}{\Gamma \vdash A} cut, E_{\neg\neg}$$

$$\left(\frac{\Gamma, A \vdash \perp}{\Gamma \vdash \neg A} I_{\neg} \right)$$

Ejemplo sin cuantificadores (2/3)

3 La convertimos a DNF

$$\begin{aligned} & \neg[((a \rightarrow b) \wedge a) \rightarrow b] \\ & \equiv \neg[\neg((a \rightarrow b) \wedge a) \vee b] && (A \rightarrow B \equiv \neg A \vee B) \\ & \equiv \neg\neg((a \rightarrow b) \wedge a) \wedge \neg b && (\neg(A \vee B) \equiv \neg A \wedge \neg B) \\ & \equiv ((a \rightarrow b) \wedge a) \wedge \neg b && (\neg\neg A \equiv A) \\ & \equiv (\neg a \vee b) \wedge a \wedge \neg b && (A \rightarrow B \equiv \neg A \vee B) \\ & \equiv (\neg a \vee b) \wedge a \wedge \neg b && ((A \vee B) \wedge C \equiv (A \wedge C) \vee (B \wedge C)) \\ & \equiv (\neg a \wedge a \wedge \neg b) \vee \\ & \quad (b \wedge a \wedge \neg b) \end{aligned}$$

Conversión a DNF - Reglas admisibles

Reglas admisibles para conversión a DNF

Pasos base

$$\neg\neg a \dashv\vdash a$$

$$\neg\perp \dashv\vdash \top$$

$$\neg\top \dashv\vdash \perp$$

$$a \rightarrow b \dashv\vdash \neg a \vee b$$

$$\neg(a \vee b) \dashv\vdash \neg a \wedge \neg b$$

$$\neg(a \wedge b) \dashv\vdash \neg a \vee \neg b$$

$$(a \vee b) \wedge c \dashv\vdash (a \wedge c) \vee (b \wedge c)$$

$$c \wedge (a \vee b) \dashv\vdash (c \wedge a) \vee (c \wedge b)$$

$$a \vee (b \vee c) \dashv\vdash (a \vee b) \vee c$$

$$a \wedge (b \wedge c) \dashv\vdash (a \wedge b) \wedge c$$

Pasos recursivos de congruencia

(con $A \dashv\vdash A'$, $B \dashv\vdash B'$)

$$A \wedge B \dashv\vdash A' \wedge B$$

$$A \wedge B \dashv\vdash A \wedge B'$$

$$A \vee B \dashv\vdash A' \vee B$$

$$A \vee B \dashv\vdash A \vee B'$$

$$\neg A \dashv\vdash \neg A'$$

¡30 demostraciones!

Ejemplo sin cuantificadores (3/3)

- 4 Refutamos cada cláusula

$$(\neg a \wedge a \wedge \neg b) \vee (b \wedge a \wedge \neg b) \vdash \perp$$



Definición (Reglas de inferencia)

$$\frac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C} \text{E}\vee$$

$$\frac{\Gamma \vdash \neg A \quad \Gamma \vdash A}{\Gamma \vdash \perp} \text{E}\neg$$

Ejemplo con cuantificadores (1/2)

By con cuantificadores

```
1 axiom ax1: forall X . p(X) -> q(X)
2 axiom ax2: p(k)
3 theorem t: q(k)
4 proof
5   thus q(k) by ax1, ax2
6 end
```


Ejemplo con cuantificadores (1/2)

By con cuantificadores

```
1 axiom ax1: forall X . p(X) -> q(X)
2 axiom ax2: p(k)
3 theorem t: q(k)
4 proof
5   thus q(k) by ax1, ax2
6 end
```

- ❶ Para certificar **thus** $q(k)$ **by** $ax1, ax2$ hay que generar una demostración para la implicación

$$\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k)$$

Ejemplo con cuantificadores (1/2)

By con cuantificadores

```
1 axiom ax1: forall X . p(X) -> q(X)
2 axiom ax2: p(k)
3 theorem t: q(k)
4 proof
5   thus q(k) by ax1, ax2
6 end
```

- ❶ Para certificar **thus** $q(k)$ **by** $ax1, ax2$ hay que generar una demostración para la implicación

$$\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k)$$

- ❷ Negamos la fórmula

$$\neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right]$$

Ejemplo con cuantificadores (2/2)

- ③ La convertimos a DNF (\forall es opaco)

$$\begin{aligned} & \neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right] \\ & \equiv (\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \wedge \neg q(k) \end{aligned}$$

Ejemplo con cuantificadores (2/2)

- ③ La convertimos a DNF (\forall es opaco)

$$\begin{aligned} & \neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right] \\ & \equiv (\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \wedge \neg q(k) \end{aligned}$$

- ④ Eliminamos $\forall x. (p(x) \rightarrow q(x))$ ($E\forall$). Reemplazamos x por una meta-variable fresca u .

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k)$$

Ejemplo con cuantificadores (2/2)

- 3 La convertimos a DNF (\forall es opaco)

$$\neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right] \\ \equiv (\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \wedge \neg q(k)$$

- 4 Eliminamos $\forall x. (p(x) \rightarrow q(x))$ ($E\forall$). Reemplazamos x por una meta-variable fresca u .

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k)$$

- 5 Convertimos a DNF

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k) \equiv (\neg p(u) \wedge p(k) \wedge \neg q(k)) \vee \\ (q(u) \wedge p(k) \wedge \neg q(k))$$

Ejemplo con cuantificadores (2/2)

- 3 La convertimos a DNF (\forall es opaco)

$$\neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right] \\ \equiv (\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \wedge \neg q(k)$$

- 4 Eliminamos $\forall x. (p(x) \rightarrow q(x))$ ($E\forall$). Reemplazamos x por una meta-variable fresca u .

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k)$$

- 5 Convertimos a DNF

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k) \equiv (\neg p(u) \wedge p(k) \wedge \neg q(k)) \vee \\ (q(u) \wedge p(k) \wedge \neg q(k))$$

- 6 Refutamos cada cláusula (con unificación).

Ejemplo con cuantificadores (2/2)

- 3 La convertimos a DNF (\forall es opaco)

$$\neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right] \\ \equiv (\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \wedge \neg q(k)$$

- 4 Eliminamos $\forall x. (p(x) \rightarrow q(x))$ ($E\forall$). Reemplazamos x por una meta-variable fresca u .

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k)$$

- 5 Convertimos a DNF

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k) \equiv (\neg p(u) \wedge p(k) \wedge \neg q(k)) \vee \\ (q(u) \wedge p(k) \wedge \neg q(k))$$

- 6 Refutamos cada cláusula (con unificación).

- $\neg p(u) \wedge p(k) \wedge \neg q(k)$ tenemos $p(u) \doteq p(k)$ con $\{u := k\}$

Ejemplo con cuantificadores (2/2)

- 3 La convertimos a DNF (\forall es opaco)

$$\neg \left[\left((\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \right) \rightarrow q(k) \right] \\ \equiv (\forall x. (p(x) \rightarrow q(x))) \wedge p(k) \wedge \neg q(k)$$

- 4 Eliminamos $\forall x. (p(x) \rightarrow q(x))$ ($E\forall$). Reemplazamos x por una meta-variable fresca u .

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k)$$

- 5 Convertimos a DNF

$$(p(u) \rightarrow q(u)) \wedge p(k) \wedge \neg q(k) \equiv (\neg p(u) \wedge p(k) \wedge \neg q(k)) \vee \\ (q(u) \wedge p(k) \wedge \neg q(k))$$

- 6 Refutamos cada cláusula (con unificación).

- $\neg p(u) \wedge p(k) \wedge \neg q(k)$ tenemos $p(u) \doteq p(k)$ con $\{u := k\}$
- $q(u) \wedge p(k) \wedge \neg q(k)$ tenemos $q(u) \doteq q(k)$ con $\{u := k\}$

Alcance y limitaciones del by

- **Completo** para lógica proposicional y **heurístico** para primer orden.
- Esto es aceptable, la validez de LPO es indecidible (Teorema de Church).
- ¿Por qué heurístico? Elimina los \forall consecutivos de a lo sumo una hipótesis (Pero le falta aún más)

- **Completo** para lógica proposicional y **heurístico** para primer orden.
- Esto es aceptable, la validez de LPO es indecidible (Teorema de Church).
- ¿Por qué heurístico? Elimina los \forall consecutivos de a lo sumo una hipótesis (Pero le falta aún más)

Ejemplo de falla en eliminación

```
1  axiom ax1: forall X . p(X) -> q(X)
2  axiom ax2: forall X . p(X)
3  theorem t: q(a)
4  proof
5      thus q(a) by ax1, ax2
6  end
```

Descarga de conjunciones

Si la tesis es una conjunción, se puede probar un subconjunto de ella y se reduce el resto.

Problema:

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \text{I}\wedge$$

Descarga

```
1  axiom "a": a
2  axiom "b": b
3  axiom "c": c
4  axiom "d": d
5  axiom "e": e
6  theorem "and discharge":
7      (a & b) & ((c & d) & e)
8  proof
9      thus a & e by "a", "e"
10     thus d by "d"
11     thus b & c by "b", "c"
12 end
```

Descarga de conjunciones

Si la tesis es una conjunción, se puede probar un subconjunto de ella y se reduce el resto.

- Reordena la conjunción (tratando como conjunto).

Descarga

$$(a \wedge e) \wedge (b \wedge c \wedge d)$$

```
1  axiom "a": a
2  axiom "b": b
3  axiom "c": c
4  axiom "d": d
5  axiom "e": e
6  theorem "and discharge":
7      (a & b) & ((c & d) & e)
8  proof
9      thus a & e by "a", "e"
10     thus d by "d"
11     thus b & c by "b", "c"
12 end
```

Descarga de conjunciones

Si la tesis es una conjunción, se puede probar un subconjunto de ella y se reduce el resto.

Descarga

```
1 axiom "a": a
2 axiom "b": b
3 axiom "c": c
4 axiom "d": d
5 axiom "e": e
6 theorem "and discharge":
7   (a & b) & ((c & d) & e)
8 proof
9   thus a & e by "a", "e"
10  thus d by "d"
11  thus b & c by "b", "c"
12 end
```

- Reordena la conjunción (tratando como conjunto).

$$(a \wedge e) \wedge (b \wedge c \wedge d)$$

- Demuestra la equivalencia con **equivalently** (por abajo, mismo solver que el by)

$$(a \wedge e) \wedge (b \wedge c \wedge d) \\ \rightarrow (a \wedge b) \wedge ((c \wedge d) \wedge e)$$

Descarga de conjunciones

Si la tesis es una conjunción, se puede probar un subconjunto de ella y se reduce el resto.

Descarga

```
1 axiom "a": a
2 axiom "b": b
3 axiom "c": c
4 axiom "d": d
5 axiom "e": e
6 theorem "and discharge":
7   (a & b) & ((c & d) & e)
8 proof
9   thus a & e by "a", "e"
10  thus d by "d"
11  thus b & c by "b", "c"
12 end
```

- Reordena la conjunción (tratando como conjunto).

$$(a \wedge e) \wedge (b \wedge c \wedge d)$$

- Demuestra la equivalencia con **equivalently** (por abajo, mismo solver que el by)

$$(a \wedge e) \wedge (b \wedge c \wedge d) \\ \rightarrow (a \wedge b) \wedge ((c \wedge d) \wedge e)$$

- **by** es completo para proposicional \Rightarrow resuelve asociatividad, conmutatividad e idempotencia (repetidos)

Extracción de testigos

Extracción simple

```
1 axiom ax: es_bajo(juan)
2 theorem t: exists Alguien . es_bajo(Alguien)
3 proof
4   take Alguien := juan
5   thus es_bajo(juan) by ax
6 end
```


Extracción simple

```
1 axiom ax: es_bajo(juan)
2 theorem t: exists Alguien . es_bajo(Alguien)
3 proof
4   take Alguien := juan
5   thus es_bajo(juan) by ax
6 end
```

take <var> := <term>

$$\frac{\Gamma \vdash A\{x := t\}}{\Gamma \vdash \exists x.A} \text{I}\exists$$

Extracción indirecta con instanciación

```
1 axiom padre_es_padre: forall A. es_padre(A, padre(A))  
2 axiom def_abuelo: forall P. forall Q. forall R.  
3   (es_padre(P, Q) & es_padre(Q, R)) <=> es_abuelo(P, R)
```

Extracción indirecta con instanciación

```
1  axiom padre_es_padre: forall A. es_padre(A, padre(A))
2  axiom def_abuelo: forall P. forall Q. forall R.
3      (es_padre(P, Q) & es_padre(Q, R)) <=> es_abuelo(P, R)
4
5  theorem todos_tienen_padre: forall Q. exists P. es_padre(Q, P)
6  proof
7      let Q
8      take P := padre(Q)
9      thus es_padre(Q, padre(Q)) by "padre_es_padre"
10 end
```

Extracción indirecta con instanciación

```
1  axiom padre_es_padre: forall A. es_padre(A, padre(A))
2  axiom def_abuelo: forall P. forall Q. forall R.
3      (es_padre(P, Q) & es_padre(Q, R)) <-> es_abuelo(P, R)
4
5  theorem todos_tienen_padre: forall Q. exists P. es_padre(Q, P)
6  proof
7      let Q
8      take P := padre(Q)
9      thus es_padre(Q, padre(Q)) by "padre_es_padre"
10 end
12 theorem todos_tienen_abuelo: forall A. exists B. es_abuelo(A, B)
13 proof
14     let A
15     consider X st "h1": es_padre(A, X) by todos_tienen_padre
16     consider Y st "h2": es_padre(X, Y) by todos_tienen_padre
17     take B := Y
18     thus es_abuelo(A, Y) by "h1", "h2", "def_abuelo"
19 end
```

Extracción indirecta

theorem todos_tienen_abuelo: **forall** A. **exists** B. es_abuelo(A, B)

```
$ ppa extract parientes.ppa \  
  --theorem todos_tienen_abuelo \  
  --terms nacho
```

Running program... OK!

Translating... OK!

Checking translated... OK!

Extracted witness: padre(padre(nacho))

of formula: es_abuelo(nacho, padre(padre(nacho)))

Extracción por el absurdo

Extracción por el absurdo

```
1 axiom juanEsBajo: bajo(juan)
2
3 theorem noTodoElMundoEsAlto: ~forall X. ~bajo(X)
4 proof
5   suppose "todosSonAltos": forall X. ~bajo(X)
6   thus false by "juanEsBajo", "todosSonAltos"
7 end
8
9 theorem hayAlguienBajo: exists X. bajo(X)
```

Extracción por el absurdo

Extracción por el absurdo

```
1 axiom juanEsBajo: bajo(juan)
2
3 theorem noTodoElMundoEsAlto: ~forall X. ~bajo(X)
4 proof
5   suppose "todosSonAltos": forall X. ~bajo(X)
6   thus false by "juanEsBajo", "todosSonAltos"
7 end
8
9 theorem hayAlguienBajo: exists X. bajo(X)
```

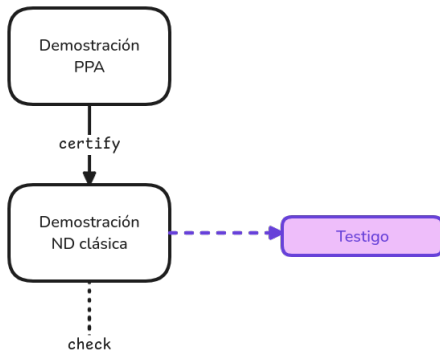
- En general $\neg\forall x.\neg\varphi \equiv \exists x.\varphi$.
- Sin **take** (\exists) explícito, igual podemos extraer el testigo a partir del **theorem** hayAlguienBajo: juan.

Extracción por el absurdo

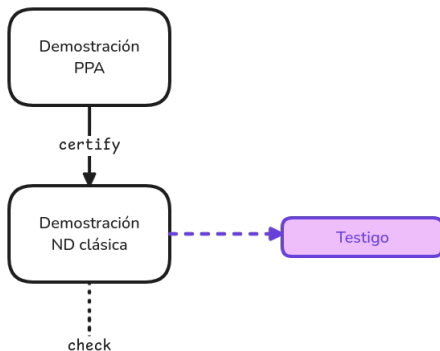
Extracción por el absurdo

```
1 axiom juanEsBajo: bajo(juan)
2
3 theorem noTodoElMundoEsAlto: ~forall X. ~bajo(X)
4 proof
5   suppose "todosSonAltos": forall X. ~bajo(X)
6   thus false by "juanEsBajo", "todosSonAltos"
7 end
8
9 theorem hayAlguienBajo: exists X. bajo(X)
```

- En general $\neg\forall x.\neg\varphi \equiv \exists x.\varphi$.
- Sin **take** (\exists) explícito, igual podemos extraer el testigo a partir del **theorem** hayAlguienBajo: juan.
- La implementación no es tan directa como buscar un \exists en el árbol de la demostración.



- Buscamos un mecanismo general que nos permita extraer testigos a partir de demostraciones en **deducción natural clásica**



- Buscamos un mecanismo general que nos permita extraer testigos a partir de demostraciones en **deducción natural clásica**
- Pero la lógica clásica **no es constructiva**, por LEM:

$$\frac{}{\Gamma \vdash A \vee \neg A} \text{LEM}$$

Ejemplo (Fórmula sin demostración constructiva)

Sea C algo indecidible (tipo HALT), queremos ver que vale

$$\exists y. (y = 1 \wedge C) \vee (y = 0 \wedge \neg C)$$

Ejemplo (Fórmula sin demostración constructiva)

Sea C algo indecidible (tipo HALT), queremos ver que vale

$$\exists y.(y = 1 \wedge C) \vee (y = 0 \wedge \neg C)$$

podemos demostrarlo por LEM, sabemos que vale $C \vee \neg C$

- Supongamos que vale C . Tomo $y = 1$.
- Supongamos que vale $\neg C$. Tomo $y = 0$.



Ejemplo (Fórmula sin demostración constructiva)

Sea C algo indecidible (tipo HALT), queremos ver que vale

$$\exists y. (y = 1 \wedge C) \vee (y = 0 \wedge \neg C)$$

podemos demostrarlo por LEM, sabemos que vale $C \vee \neg C$

- Supongamos que vale C . Tomo $y = 1$.
- Supongamos que vale $\neg C$. Tomo $y = 0$.



¡No nos dice explícitamente si $y = 1$ o $y = 0$! No es *constructiva*.

Demostración no constructiva

Ejemplo (Fórmula sin demostración constructiva)

Sea C algo indecidible (tipo HALT), queremos ver que vale

$$\exists y. (y = 1 \wedge C) \vee (y = 0 \wedge \neg C)$$

podemos demostrarlo por LEM, sabemos que vale $C \vee \neg C$

- Supongamos que vale C . Tomo $y = 1$.
- Supongamos que vale $\neg C$. Tomo $y = 0$.

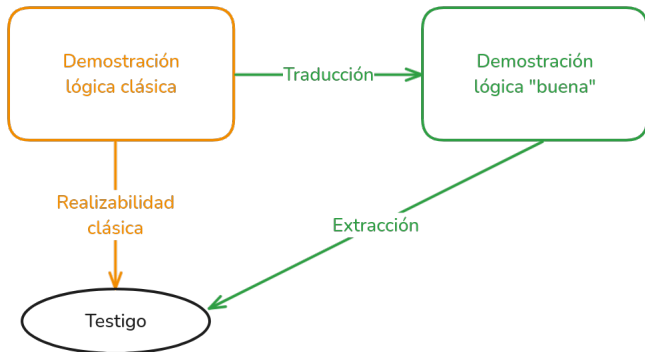


¡No nos dice explícitamente si $y = 1$ o $y = 0$! No es *constructiva*.

¿Entonces por qué lógica clásica?

- Existen fórmulas que admiten demostraciones constructivas y no constructivas, y otras *solo no constructivas* (i.e. clásicas) Ejemplo: $\neg(A \wedge B) \rightarrow \neg A \vee B$ solo es válido en lógica clásica.
- Además permite razonar por el absurdo, con $E\neg\neg \equiv \text{LEM}$.

Clases de estrategias de extracción



Clases de estrategias de extracción de demostraciones en lógica clásica:

- **Directas:** Extraer directamente de demostraciones clásicas. Técnicas de *realizabilidad clásica* (Semánticas de λ -cálculos clásicos).
- **Indirectas:** Convertir la demostración a una lógica que se porte mejor y extraer de ahí.

lógica intuicionista = lógica clásica – LEM

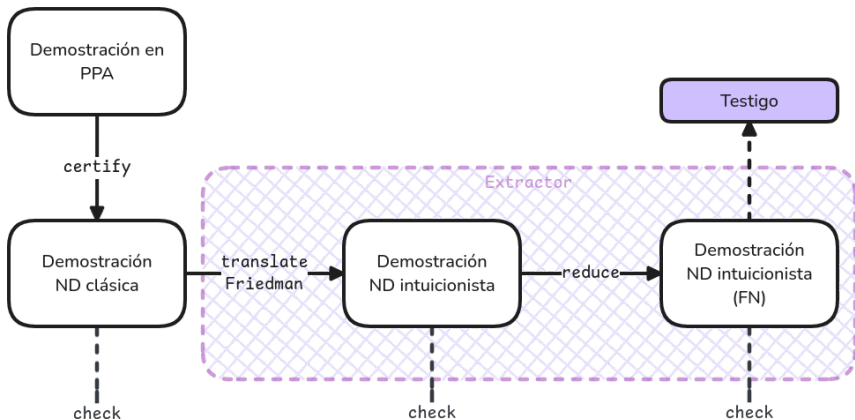
Características:

- No tiene LEM³, entonces siempre es constructiva.
- Siempre permite hacer extracción de testigos: proceso de normalización con *forma normal* buena, una demostración de un \exists debería comenzar con $I\exists$ y de ahí sacás el testigo.

$$\frac{\Gamma \vdash A\{x := t\}}{\Gamma \vdash \exists x.A} I\exists$$

³Ni principios de razonamiento equivalentes, como $E\neg\neg$

Estrategia de extracción indirecta



Traducción de doble negación relativizada

Definición (Traducción de doble negación relativizada)

Podemos ver $\neg A \equiv A \rightarrow \perp$. Definimos $\neg_R A \equiv A \rightarrow R$,

$$\perp^{\neg\neg} = R$$

$$A^{\neg\neg} = \neg_R \neg_R A \quad \text{con } A \text{ atómica}$$

$$(\neg A)^{\neg\neg} = \neg_R A^{\neg\neg}$$

$$(A \wedge B)^{\neg\neg} = A^{\neg\neg} \wedge B^{\neg\neg}$$

$$(A \vee B)^{\neg\neg} = \neg_R (\neg_R A^{\neg\neg} \wedge \neg_R B^{\neg\neg})$$

$$(A \rightarrow B)^{\neg\neg} = A^{\neg\neg} \rightarrow B^{\neg\neg}$$

$$(\forall x. A)^{\neg\neg} = \forall x. A^{\neg\neg}$$

$$(\exists x. A)^{\neg\neg} = \neg_R \forall x. \neg_R A^{\neg\neg}$$

- Método general para embeber clásica en intuicionista.
- En clásica son equivalentes ($E\neg$).
- En intuicionista es más débil.

Teorema

Si $\Pi \triangleright \Gamma \vdash_C A$, luego $\Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I A^{\neg\neg}$

Traducción de doble negación relativizada

Definición (Traducción de doble negación relativizada)

Podemos ver $\neg A \equiv A \rightarrow \perp$. Definimos $\neg_R A \equiv A \rightarrow R$,

$$\perp^{\neg\neg} = R$$

$$A^{\neg\neg} = \neg_R \neg_R A \quad \text{con } A \text{ atómica}$$

$$(\neg A)^{\neg\neg} = \neg_R A^{\neg\neg}$$

$$(A \wedge B)^{\neg\neg} = A^{\neg\neg} \wedge B^{\neg\neg}$$

$$(A \vee B)^{\neg\neg} = \neg_R (\neg_R A^{\neg\neg} \wedge \neg_R B^{\neg\neg})$$

$$(A \rightarrow B)^{\neg\neg} = A^{\neg\neg} \rightarrow B^{\neg\neg}$$

$$(\forall x. A)^{\neg\neg} = \forall x. A^{\neg\neg}$$

$$(\exists x. A)^{\neg\neg} = \neg_R \forall x. \neg_R A^{\neg\neg}$$

- Método general para embeber clásica en intuicionista.
- En clásica son equivalentes ($E\neg$).
- En intuicionista es más débil.

Teorema

Si $\Pi \triangleright \Gamma \vdash_C A$, luego $\Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I A^{\neg\neg}$

Definición (Fórmulas conjuntivas)

$$C ::= \perp \mid \top \mid p(t_1, \dots, t_n) \mid C \wedge C$$

Teorema (Traducción de Friedman)

Sea φ una fórmula **conjuntiva**. Si tenemos

$$\Pi \triangleright \Gamma \vdash_C \forall y_1 \dots \forall y_n. \exists x. \varphi(x, y_1, \dots, y_n),$$

podemos generar una demostración intuicionista de la misma fórmula.

Definición (Fórmulas conjuntivas)

$$C ::= \perp \mid \top \mid p(t_1, \dots, t_n) \mid C \wedge C$$

Teorema (Traducción de Friedman)

Sea φ una fórmula **conjuntiva**. Si tenemos

$$\Pi \triangleright \Gamma \vdash_C \forall y_1 \dots \forall y_n. \exists x. \varphi(x, y_1, \dots, y_n),$$

podemos generar una demostración intuicionista de la misma fórmula.

Lema (Traducción de Friedman simplificada)

Idem con $\Pi \triangleright \Gamma \vdash_C \exists x. \varphi$,

Demostración.

Aplicando la traducción con $R = \exists x.\varphi$, tenemos que

$$(\Pi \triangleright \Gamma \vdash_C \exists x.\varphi)^{\neg\neg} \Leftrightarrow \Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I \neg_R \forall x.\neg_R \varphi^{\neg\neg}$$

Luego,

El truco de Friedman

Demostración.

Aplicando la traducción con $R = \exists x.\varphi$, tenemos que

$$(\Pi \triangleright \Gamma \vdash_C \exists x.\varphi)^{\neg\neg} \Leftrightarrow \Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I \neg_R \forall x. \neg_R \varphi^{\neg\neg}$$

Luego,

$$\frac{\Pi^{\neg\neg} \quad \frac{\frac{\frac{\overline{\Gamma^{\neg\neg}, \varphi \vdash_I \varphi} \text{Ax}}{\Gamma^{\neg\neg}, \varphi \vdash_I R = \exists x.\varphi} \text{I}\exists}{\Gamma^{\neg\neg} \vdash_I \neg_R \varphi} \text{I}\rightarrow}{\Gamma^{\neg\neg} \vdash_I \neg_R \varphi^{\neg\neg}} \text{cut, I}(\neg_R \cdot \neg\neg)}{\Gamma^{\neg\neg} \vdash_I \forall x \neg_R \varphi^{\neg\neg}} \text{I}\forall}{\Gamma^{\neg\neg} \vdash_I \exists x.\varphi} \text{E}\rightarrow$$



Lema (Introducción de \neg_R)

Si A es conjuntiva, entonces vale $\neg_R A \vdash_I \neg_R A^{\neg\neg}$ y lo notamos con la regla admisible $I(\neg_R \cdot \neg\neg)$.

Introducción de negación relativizada

Lema (Introducción de \neg_R)

Si A es conjuntiva, entonces vale $\neg_R A \vdash_I \neg_R A^{\neg\neg}$ y lo notamos con la regla admisible $I(\neg_R \cdot \neg\neg)$.

Demostración.

Por inducción estructural en la fórmula.

- Atómicas trivial (y eso aparece en la bib)
- Conjunción tiene algunos trucos.



Teorema

Si $\Pi \triangleright \Gamma \vdash_C A$, luego $\Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I A^{\neg\neg}$

Teorema

Si $\Pi \triangleright \Gamma \vdash_C A$, luego $\Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I A^{\neg\neg}$

Demostración.

Inducción estructural sobre la demostración.

Estrategia: traducir recursivamente las partes de Π y usarlas para construir una nueva demostración de $A^{\neg\neg}$. □

Traducción de introducción de conjunción

Lema (Traducción de I_{\wedge})

$$\frac{\Pi_A \quad \Gamma \vdash_C A \quad \Pi_B \quad \Gamma \vdash_C B}{\Gamma \vdash_C A \wedge B} I_{\wedge}$$

Es posible demostrar
 $(A \wedge B)^{\neg\neg} = A^{\neg\neg} \wedge B^{\neg\neg}$.

Demostración.

Usando la HI:

$$\Pi_A^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I A^{\neg\neg} \text{ y } \Pi_B^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I B^{\neg\neg},$$

$$\frac{\Pi_A^{\neg\neg} \quad \Gamma^{\neg\neg} \vdash_I A^{\neg\neg} \quad \Pi_B^{\neg\neg} \quad \Gamma^{\neg\neg} \vdash_I B^{\neg\neg}}{\Gamma^{\neg\neg} \vdash_I A^{\neg\neg} \wedge B^{\neg\neg}} I_{\wedge}$$



Teorema

Si $\Pi \triangleright \Gamma \vdash_C A$, luego $\Pi^{\neg\neg} \triangleright \Gamma^{\neg\neg} \vdash_I A^{\neg\neg}$

Demostración.

Inducción estructural sobre la demostración.

Estrategia: traducir recursivamente las partes de Π y usarlas para construir una nueva demostración de $A^{\neg\neg}$.

- $I\wedge$, $E\wedge_1$, $E\wedge_2$, $I\rightarrow$, $E\rightarrow$, $I\vee_1$, $I\vee_2$, $I\forall$, $E\forall$, $I\neg$, $E\neg$, $I\top$, Ax , $I\exists$ directas.
- **LEM** interesante.
- $E\perp$ inducción estructural sobre la fórmula.
- $E\vee$ y $E\exists$ análogos. Truco: No vale $E\neg\neg$ pero si $E\neg_{R\neg R}$ (probado por inducción estructural sobre la fórmula).



Lema (Traducción de Friedman simplificada)

Sea φ una fórmula conjuntiva. Si tenemos $\Gamma \vdash_C \exists x.\varphi$, podemos generar una demostración intuicionista de la misma fórmula $\Gamma^{\neg\neg} \vdash_I \exists x.\varphi$.

Lema (Traducción de Friedman simplificada)

Sea φ una fórmula conjuntiva. Si tenemos $\Gamma \vdash_C \exists x.\varphi$, podemos generar una demostración intuicionista de la misma fórmula $\Gamma^{\neg\neg} \vdash_I \exists x.\varphi$.

Problema: la demostración normalizada no puede comenzar con \exists

$$p(v)^{\neg\neg} \vdash_I \exists x.p(x) \iff \neg_R \neg_R p(v) \vdash_I \exists x.p(x)$$

Nos gustaría *mantener el contexto original*: $p(v) \vdash_I \exists x.p(x)$

Problema con axiomas

Lema (Traducción de Friedman simplificada)

Sea φ una fórmula conjuntiva. Si tenemos $\Gamma \vdash_C \exists x.\varphi$, podemos generar una demostración intuicionista de la misma fórmula $\Gamma^{\neg\neg} \vdash_I \exists x.\varphi$.

Problema: la demostración normalizada no puede comenzar con \exists

$$p(v)^{\neg\neg} \vdash_I \exists x.p(x) \iff \neg_R \neg_R p(v) \vdash_I \exists x.p(x)$$

Nos gustaría *mantener el contexto original*: $p(v) \vdash_I \exists x.p(x)$

Manteniendo el contexto

Luego de la traducción, antes de reducir, reemplazamos cada cita (Ax) de un axioma $h : \varphi^{\neg\neg}$ por la demostración $\varphi \vdash_I \varphi^{\neg\neg}$.

Lema (Introducción de la traducción $\neg\neg$)

Si φ es una F-fórmula, vale $\varphi \vdash_I \varphi^{\neg\neg}$.

F-fórmulas

$$A ::= \perp \mid \top \mid p(t_1, \dots, t_n)$$
$$F ::= A$$
$$\mid F \wedge F \mid F \vee F$$
$$\mid \forall x.F \mid \exists x.F$$
$$\mid C \rightarrow F \mid \neg C$$
$$C ::= A \mid C \wedge C$$

- A : Fórmulas atómicas
- F : F-fórmulas
- C : Fórmulas conjuntivas

F-fórmulas

$$A ::= \perp \mid \top \mid p(t_1, \dots, t_n)$$
$$F ::= A$$
$$\mid F \wedge F \mid F \vee F$$
$$\mid \forall x.F \mid \exists x.F$$
$$\mid C \rightarrow F \mid \neg C$$
$$C ::= A \mid C \wedge C$$

- A : Fórmulas atómicas
- F : F-fórmulas
- C : Fórmulas conjuntivas

Fórmulas de Harrop

$$G ::= A$$
$$\mid G \wedge G \mid G \vee G$$
$$\mid \forall x.G \mid \exists x.G$$
$$\mid H \rightarrow G$$
$$H ::= A \mid H \wedge H$$
$$\mid \forall x.H$$
$$\mid G \rightarrow A$$

- G : G-fórmulas
- H : Fórmulas Harrop Hereditarias
- Generalización de cláusulas de Horn, usadas para realizabilidad

Motivación: evitar “desvíos superfluos”.

Ejemplo

$$\frac{\frac{\frac{A \vdash A}{\vdash A \rightarrow A} \text{Ax}}{\vdash (A \rightarrow A) \wedge (B \rightarrow B)} \text{I} \rightarrow \quad \frac{\frac{B \vdash B}{\vdash B \rightarrow B} \text{Ax}}{\vdash (A \rightarrow A) \wedge (B \rightarrow B)} \text{I} \wedge}{\vdash A \rightarrow A} \text{E} \wedge_1 \quad \rightsquigarrow \quad \frac{\frac{A \vdash A}{\vdash A \rightarrow A} \text{Ax}}{\vdash A \rightarrow A} \text{I} \rightarrow$$

Motivación: evitar “desvíos superfluos”.

Ejemplo

$$\frac{\frac{\overline{A \vdash A}^{Ax}}{\vdash A \rightarrow A} I \rightarrow \quad \frac{\overline{B \vdash B}^{Ax}}{\vdash B \rightarrow B} I \rightarrow}{\vdash (A \rightarrow A) \wedge (B \rightarrow B)} I \wedge \quad \rightsquigarrow \quad \frac{\overline{A \vdash A}^{Ax}}{\vdash A \rightarrow A} I \rightarrow$$

$E \wedge_1$

- Se van a ver todos de esa forma: Una **eliminación** demostrada inmediatamente por su **introducción** correspondiente.
- Ejemplo: $E \wedge_1$ demostrada por $I \wedge$.
- Idea: Simplificarlos sucesivamente hasta que no haya más y esté en **forma normal**.

- **Isomorfismo Curry-Howard:** correspondencia entre demostraciones en deducción natural y términos de λ -cálculo.
- Normalización de demostraciones corresponde a semántica de λ -cálculo. Puede ser más intuitivo en cálculos.

Ejemplo

Conjunciones como el tipo de las tuplas, y las eliminaciones como proyecciones.

$$\pi_1(\langle M_1, M_2 \rangle) \rightsquigarrow M_1$$

$$\pi_2(\langle M_1, M_2 \rangle) \rightsquigarrow M_2$$

$$\frac{\frac{\frac{\Pi_1}{\Gamma \vdash A_1} \quad \frac{\Pi_2}{\Gamma \vdash A_2}}{\Gamma \vdash A_1 \wedge A_2} I_{\wedge} \quad \rightsquigarrow \quad \frac{\Pi_i}{\Gamma \vdash A_i} E_{\wedge_i}}{\Gamma \vdash A_i} E_{\wedge_i}$$

Definición (Normalización de implicación)

$$\frac{\frac{\frac{\Pi_B}{\Gamma, h : A \vdash B}}{\Gamma \vdash A \rightarrow B} \mid \rightarrow_h \quad \frac{\Pi_A}{\Gamma \vdash A} \text{E} \rightarrow}{\Gamma \vdash B} \rightsquigarrow \frac{\Pi_B}{\Gamma \vdash B}$$

- Primer idea: $\Pi_B \triangleright \Gamma \vdash B$

Normalización de implicación

Definición (Normalización de implicación)

$$\frac{\frac{\frac{\Pi_B}{\Gamma, h : A \vdash B} \text{I} \rightarrow_h}{\Gamma \vdash A \rightarrow B} \quad \frac{\Pi_A}{\Gamma \vdash A} \text{E} \rightarrow}{\Gamma \vdash B} \rightsquigarrow \frac{\Pi_B \{h := \Pi_A\}}{\Gamma \vdash B}$$

- Primer idea: ~~$\Pi_B \triangleright \Gamma \vdash B$~~
- Π_B requiere $h : A$, agregada por $\text{I} \rightarrow_h$
- Correcto: usar Π_B , pero *sustituyendo* todas las ocurrencias de la hipótesis h por la demostración Π_A (sin capturas).

Normalización de implicación

Definición (Normalización de implicación)

$$\frac{\frac{\frac{\Pi_B}{\Gamma, h : A \vdash B} \mid \rightarrow_h}{\Gamma \vdash A \rightarrow B} \quad \frac{\Pi_A}{\Gamma \vdash A} \quad E \rightarrow}{\Gamma \vdash B} \rightsquigarrow \frac{\Pi_B \{h := \Pi_A\}}{\Gamma \vdash B}$$

- Primer idea: ~~$\Pi_B \triangleright \Gamma \vdash B$~~
- Π_B requiere $h : A$, agregada por $\mid \rightarrow_h$
- Correcto: usar Π_B , pero *sustituyendo* todas las ocurrencias de la hipótesis h por la demostración Π_A (sin capturas).

Definición (Otras reglas)

Además, hay reglas para

- $E\exists$ con $I\exists$, $E\forall$ con $I\forall$.
- $E\neg$ con $I\neg$, $E\vee$ con $I\vee$.

Algoritmo de reducción

Idea original: reducir en un paso sucesivamente hasta que sea irreducible.

$$\begin{array}{ccccccc} \Pi_A & \Pi_B & \Pi_A^1 & \Pi_B & \dots & \Pi_B & \Pi_A^* & \Pi_B^1 \\ \vdots & & \vdots & & & \vdots & & \vdots \\ \Pi & \rightsquigarrow & \Pi & & & \Pi & \rightsquigarrow & \Pi \end{array}$$

Algoritmo de reducción

Idea original: reducir en un paso sucesivamente hasta que sea irreducible.

$$\begin{array}{ccccccc} \Pi_A & \Pi_B & \Pi_A^1 & \Pi_B & \dots & \Pi_B & \Pi_A^* & \Pi_B^1 \\ \vdots & & & \vdots & & \vdots & & \vdots \\ \Pi & \rightsquigarrow & \Pi & & & \Pi & \rightsquigarrow & \Pi \end{array}$$

Problema: Muy lento (demostraciones muy grandes). Estrategias alternativas:

- Un paso o muchos pasos
- **Gross Knuth:** reduce en muchos pasos todos los sub-términos posibles al mismo tiempo.

Algoritmo de reducción

Idea original: reducir en un paso sucesivamente hasta que sea irreducible.

$$\begin{array}{ccccccc} \Pi_A & \Pi_B & \Pi_A^1 & \Pi_B & \Pi_A^* & \Pi_B & \Pi_A^* & \Pi_B^1 \\ \vdots & & \vdots & & \vdots & & \vdots & \\ \Pi & & \Pi & & \Pi & & \Pi & \end{array} \rightsquigarrow \dots \rightsquigarrow$$

Problema: Muy lento (demostraciones muy grandes). Estrategias alternativas:

- Un paso o muchos pasos
- **Gross Knuth:** reduce en muchos pasos todos los sub-términos posibles al mismo tiempo.

En un solo paso,

$$\begin{array}{ccc} \Pi_A & \Pi_B & \Pi_A^* \Pi_B^* \\ \vdots & & \vdots \\ \Pi & & \Pi \end{array} \rightsquigarrow$$

- **Incompleta:** no contempla *reducciones permutativas* (mezclando introducciones y eliminaciones de conectivos distintos).

- **Incompleta:** no contempla *reducciones permutativas* (mezclando introducciones y eliminaciones de conectivos distintos).
 - *Mejora:* Implementarlas.

- **Incompleta:** no contempla *reducciones permutativas* (mezclando introducciones y eliminaciones de conectivos distintos).
 - *Mejora:* Implementarlas.
- **Ineficiente:** en cada paso reinicia la búsqueda de todos los focos de evaluación.

- **Incompleta:** no contempla *reducciones permutativas* (mezclando introducciones y eliminaciones de conectivos distintos).
 - *Mejora:* Implementarlas.
- **Ineficiente:** en cada paso reinicia la búsqueda de todos los focos de evaluación.
 - *Mejora:* Usar una máquina abstracta que implemente reducción a forma normal, Crégut para reducción *call-by-name* fuerte o la máquina de Biernacka para reducción *call-by-need* fuerte.

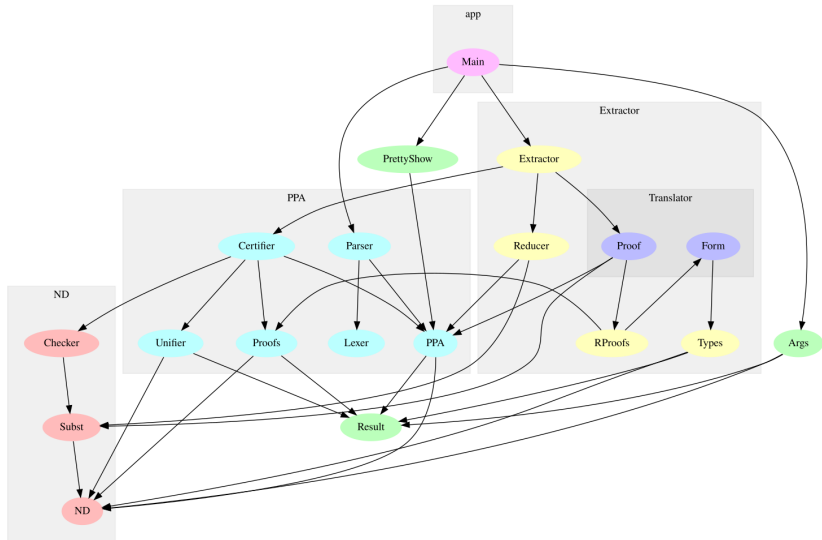
Programa con falla de extracción

```
1 axiom ax_1: roba(tuco) | mata(tuco)
2 axiom ax_2: forall X . roba(X) -> criminal(X)
3 axiom ax_3: forall X . mata(X) -> criminal(X)
4
5 theorem t: exists X . criminal(X)
6 proof
7   take X := tuco
8   cases by ax_1
9     case roba(tuco)
10       hence criminal(tuco)
11         by ax_2
12
13     case mata(tuco)
14       hence criminal(tuco)
15         by ax_3
16   end
17 end
```

Certifica el programa generando una demostración que en lugar de comenzar con \exists , comienza con $\exists V$ y en cada rama introduce el existencial dos veces, con el mismo término

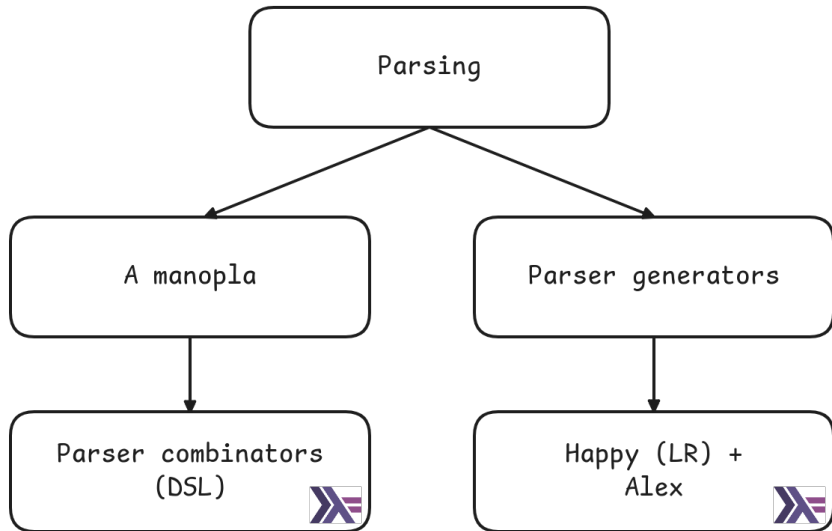
Detalles de implementación

La herramienta ppa



Haskell, 19 módulos con 330 tests

String -> Estructura



Trabajo futuro

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.
- Refinar fórmulas conjuntivas. Profundizar vínculo con Harrop.

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.
- Refinar fórmulas conjuntivas. Profundizar vínculo con Harrop.
- Sofisticar reducción de demostraciones: hacer completa (reglas permutativas) y más eficiente (implementando máquina abstracta).

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.
- Refinar fórmulas conjuntivas. Profundizar vínculo con Harrop.
- Sofisticar reducción de demostraciones: hacer completa (reglas permutativas) y más eficiente (implementando máquina abstracta).
- Mejorar PPA como lenguaje de programación: módulos, importar archivos, biblioteca estándar.

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.
- Refinar fórmulas conjuntivas. Profundizar vínculo con Harrop.
- Sofisticar reducción de demostraciones: hacer completa (reglas permutativas) y más eficiente (implementando máquina abstracta).
- Mejorar PPA como lenguaje de programación: módulos, importar archivos, biblioteca estándar.
- Extender PPA con tipos (usando LPO *many-sorted* con géneros).

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.
- Refinar fórmulas conjuntivas. Profundizar vínculo con Harrop.
- Sofisticar reducción de demostraciones: hacer completa (reglas permutativas) y más eficiente (implementando máquina abstracta).
- Mejorar PPA como lenguaje de programación: módulos, importar archivos, biblioteca estándar.
- Extender PPA con tipos (usando LPO *many-sorted* con géneros).
- Modelar de forma nativa inducción (segundo orden) e igualdad.

- Sofisticar el *solver heurístico* del **by** (recursivo, eliminar más de una hipótesis).
- Extender traducción de Friedman a más de un existencial.
- Refinar fórmulas conjuntivas. Profundizar vínculo con Harrop.
- Sofisticar reducción de demostraciones: hacer completa (reglas permutativas) y más eficiente (implementando máquina abstracta).
- Mejorar PPA como lenguaje de programación: módulos, importar archivos, biblioteca estándar.
- Extender PPA con tipos (usando LPO *many-sorted* con géneros).
- Modelar de forma nativa inducción (segundo orden) e igualdad.
- Mejorar reporte de errores (muy bajo nivel).

- QR con la página
- Preguntas