PPA

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

Manuel Panichelli

Deparatamento de Computación, FCEyN, UBA

Diciembre 2024

Introducción

Asistentes de demostración

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

[1] Terrence Tao - Machine Assisted Proof

Asistentes de demostración

Implementan distintas *teorías*. (TODO: No me gusta teoria, se usa para teorias de primer orden) Ejemplos:

- Mizar (lógica de primer orden)
- Coq (teoría de tipos)
- Agda (teoría de tipos)
- Isabelle (lógica de orden superior / teoría de conjuntos ZF)

Representación de demostraciones

Queremos escribir demostraciones en la computadora. ¿Cómo las representamos?. Veamos un ejemplo.

- Tenemos dos premisas
  1. Los alumnos que faltan a los exámenes, los reprueban.
  2. Si se reprueba un final, se recursa la materia.
- Con estas dos, podríamos demostrar que si un alumno falta a un final, entonces recursa la materia.

Teorema

Si ((falta entonces reprueba) y (reprueba entonces recursa)) y falta, entonces recursa

Demostración

- Asumo que falta. Quiero ver que recursa.
- Sabemos que si falta, entonces reprueba. Por lo tanto reprobó.
- Sabemos que si reprueba, entonces recursa. Por lo tanto recursó. $\square$

Sistemas deductivos

- La demostración anterior es poco precisa. No se puede representar rigurosamente.
- Necesitamos **sistemas deductivos**: sistemas lógicos formales usados para demostrar setencias. Pueden ser representados como un tipo abstracto de datos.
- Usamos **deducción natural**. Compuesto por,
  - **Lenguaje formal**: lógica de primer orden.
  - **Reglas de inferencia**: lista de reglas que se usan para probar teoremas a partir de axiomas y otros teoremas. Por ejemplo, *modus ponens* (si es cierto $A \rightarrow B$ y A, se puede concluir B) o *modus tollens* (si es cierto $A \rightarrow B$ y $\neg B$, se puede concluir $\neg A$)
  - **Axiomas**: fórmulas de L que se asumen válidas. Todos los teoremas se derivan de axiomas. Se usan para modelar *teorías* de primer orden (por ej. teoría de estudiantes en la facultad).

Lógica de primer orden

Definición (Términos)

Los términos están dados por la gramática:

$$\begin{aligned} t ::= {} & x && \text{(variables)} \\ & \mid f(t_1, \ldots, t_n) && \text{(funciones)} \end{aligned}$$

Definición (Fórmulas)

Las fórmulas están dadas por la gramática:

$$\begin{aligned} A, B ::= {} & p(t_1, \ldots, t_n) && \text{(predicados)} \\ & \mid \bot \mid \top && \text{(falso o \textit{bottom} y verdadero o \textit{top})} \\ & \mid A \wedge B \mid A \vee B && \text{(conjunción y disyunción)} \\ & \mid A \to B \mid \neg A && \text{(implicación y negación)} \\ & \mid \forall x.A \mid \exists x.A && \text{(cuantificador universal y existencial)} \end{aligned}$$

Los predicados son **fórmulas atómicas**. Los de aridad 0 además son llamados *variables proposicionales*.

Notación

Usamos

- $x, y, z, \ldots$ como **variables**.
- $f, g, h, \ldots$ como **símbolos de función**.
- $p, q, r, \ldots$ como **símbolos de predicado**.
- $t, u, \ldots$ para referirnos a **términos**.
- $a, b, c, \ldots, A, B, C, \ldots$ y $\varphi, \psi, \ldots$ para referirnos a **fórmulas**.

Deducción natural

Deducción natural

Definiciones

- Γ es un **contexto de demostración**, conjunto de fórmulas que se asumen válidas
- Notación: $\Gamma, \varphi = \Gamma \cup \{\varphi\}$
- $\vdash$ es la **relación de derivabilidad** definida a partir de las *reglas de inferencia*. Permite escribir juicios $\Gamma \vdash \varphi$.
- Intuición: *"φ es una consecuencia de las suposiciones de Γ"*
- El juicio es cierto si en una cantidad finita de pasos podemos concluir φ a partir de las fórmulas de Γ, los axiomas y las reglas de inferencia.
- Decimos que φ es *derivable* a partir de Γ.

Reglas de inferencia

Definición (Reglas de inferencia)

$$\frac{}{\Gamma, A \vdash A}\ \mathrm{Ax} \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \mathrm{I}\wedge$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ \mathrm{I}{\to} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \mathrm{E}\wedge_1$$

$$\frac{\Gamma \vdash A \to B \quad \Gamma \vdash A}{\Gamma \vdash B}\ \mathrm{E}{\to} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \mathrm{E}\wedge_2$$

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

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

Ejemplo

Vamos a demostrar el ejemplo informal en deducción natural. Lo modelamos para un alumno y materia particulares. Notamos:

- $X \equiv \mathrm{reprueba}(\mathit{juan}, \mathrm{final}(\mathit{logica}))$
- $R \equiv \mathrm{recursa}(\mathit{juan}, \mathit{logica})$
- $F \equiv \mathrm{falta}(\mathit{juan}, \mathrm{final}(\mathit{logica}))$

Queremos probar entonces

$$\big((F \to X) \land (X \to R)\big) \to (F \to R)$$

Ejemplo

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash (F \to X) \wedge (X \to R)}\,\mathsf{Ax}}{\Gamma \vdash X \to R}\,\mathsf{E}\wedge_1 \qquad \begin{matrix}\Pi \\ \Gamma \vdash X\end{matrix}}{\Gamma = (F \to X) \wedge (X \to R),\ F \vdash R}\,\mathsf{E}{\to}}{\dfrac{(F \to X) \wedge (X \to R) \vdash F \to R}{\vdash \Big((F \to X) \wedge (X \to R)\Big) \to (F \to R)}\,\mathsf{I}{\to}}\,\mathsf{I}{\to}$$

donde

$$\Pi = \dfrac{\dfrac{\dfrac{}{\Gamma \vdash (F \to X) \wedge (X \to R)}\,\mathsf{Ax}}{\Gamma \vdash F \to X}\,\mathsf{E}\wedge_2 \qquad \dfrac{}{\Gamma \vdash F}\,\mathsf{Ax}}{\Gamma \vdash X}\,\mathsf{E}{\to}$$

Reglas de inferencia

Definiciones

Introducción

- PPA (*Pani's proof assistant*) es un **asistente de demostraciones** inspirado en Mizar.
- Es un **lenguaje de programación** implementado en Haskell que permite escribir y chequear demostraciones en lógica *clásica* de primer órden.
- A diferencia de Prolog, **no demuestra todo automáticamente***. Deben ser escritas *rigurosamente* por el usuario.
- (WIP) permite la extracción de testigos mediante la **traducción de Friedman**.

Asistentes de demostraciones (*proof assistants*)

- Son programas que *asisten* al usuario a la hora de escribir demostraciones, permiten representarlas en un programa
- Aplicaciones: Formalización de teoremas, verificación formal de programas, etc.
- Ejemplos: Coq, Isabelle (Isar), **Mizar**, ...
- Ventajas[2]:
  - facilitan colaboración a gran escala (via confianza en el checker)
  - habilitan generación automática de demostraciones con ML. Un LLM suele devolver alucinaciones, pero pueden ser chequeadas

[2] Terrence Tao - Machine Assisted Proof

Arquitectura de PPA

¿Por qué certificados?

- Si formalizamos una demostración en PPA y queremos chequear que sea correcta, hay que confiar en la implementación del *proof assistant*.
- **Criterio de De Brujin**: si guardamos una demostración de bajo nivel de forma completa, puede ser chequeada por un programa independiente (que es sencillo de implementar).
- Cumplida por Coq, pero no Mizar[3].

[3] Adam Naumowicz - A brief overview of Mizar

Deducción natural

- Los certificados emitidos por PPA son demostraciones en **deducción natural**.
- Es un sistema lógico que nos permite construir demostraciones mediante reglas de inferencia
- Estas reglas definen la relación $\Gamma \vdash \varphi$.
  Intuición: "φ es una consecuencia de las suposiciones de Γ"

Deducción natural

Reglas

$$\frac{}{\Gamma, A \vdash A}\ \mathrm{Ax} \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \mathrm{I}\wedge$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ \mathrm{I}\to \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \mathrm{E}\wedge_1$$

$$\frac{\Gamma \vdash A \to B \quad \Gamma \vdash A}{\Gamma \vdash B}\ \mathrm{E}\to \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \mathrm{E}\wedge_2$$

Dos tipos de regla:

- **introducción**: ¿cómo lo demuestro?
- **eliminación**: ¿cómo lo uso para demostrar otra cosa?

Deducción natural

Demostración de ejemplo

$$
\dfrac{
  \dfrac{
    \dfrac{}{\Gamma \vdash (A \to B) \wedge (B \to C)}\,\mathrm{Ax}
  }{\Gamma \vdash B \to C}\,\mathrm{E}\wedge_2
  \qquad
  \dfrac{
    \dfrac{\vdots}{\Gamma \vdash A \to B}\,\mathrm{E}\wedge_1
    \qquad
    \dfrac{}{\Gamma \vdash A}\,\mathrm{Ax}
  }{\Gamma \vdash B}\,\mathrm{E}{\to}
}{
  \dfrac{
    {\color{orange}\Gamma =} \big((A \to B) \wedge (B \to C)\big), A \vdash C
  }{
    \dfrac{\big((A \to B) \wedge (B \to C)\big) \vdash A \to C}{\vdash \big((A \to B) \wedge (B \to C)\big) \to (A \to C)}\,\mathrm{I}{\to}
  }\,\mathrm{I}{\to}
}\,\mathrm{E}{\to}
$$

Deducción natural

Reglas de cuantificadores

$$\frac{\Gamma \vdash A \qquad x \notin fv(\Gamma)}{\Gamma \vdash \forall x.A}\ \mathrm{I}\forall$$

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

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

$$\frac{\Gamma \vdash \exists x.A \qquad \Gamma, A \vdash B \qquad x \notin fv(\Gamma, B)}{\Gamma \vdash B}\ \mathrm{E}\exists$$

PPA

- Es un lenguaje. Frontend implementado con un *parser generator* (happy + alex)
- Permite definir axiomas y teoremas con sus demostraciones, que al **certificarse** generan una demostración en deducción natural.
- Basado en *Mathematical Vernacular*[4]: un leguaje formal para escribir demostraciones similar al natural.

[4] The Mathematical Vernacular - Freek Wiedijk

Ejemplo

Una demostración es una secuencia de *comandos*, que pueden ir sucesivamente reduciendo la *tesis* (objetivo a probar) y agregando hipótesis a un contexto. Se mapean a reglas de deducción natural.

Teorema

```
theorem "implication transitivity":
    (a -> b) & (b -> c) -> (a -> c) // Tesis
proof
    suppose h1: (a -> b) & (b -> c)
    // Tesis: a -> c

    suppose h2: a
    // Tesis: c

    thus c by h1, h2
end
```

by

- El mecanismo principal para demostrar es el **by**, que *automáticamente* demuestra que un hecho es consecuencia de una lista de hipótesis.
- Se usa en lugar de E$\rightarrow$ y E$\forall$
- Es completo para lógica proposicional pero heurístico para LPO

Ejemplo

```
axoim ax1: a -> b
axiom ax2: a
theorem thm: b
proof
    thus b by ax1, ax2
end
```

Para demostrar $((a \to b) \wedge a) \to b$ lo hacemos por el absurdo: negamos y encontramos una contradicción.

DNF

Primero convertimos la fórmula a forma normal disyuntiva (DNF)

$$\begin{aligned}
&\neg\big[\big((a \to b) \land a\big) \to b\big] && \\
&\equiv \neg\big[\neg\big((a \to b) \land a\big) \lor b\big] && (x \to y \equiv \neg x \lor y) \\
&\equiv \neg\neg\big((a \to b) \land a\big) \land \neg b && (\neg(x \lor y) \equiv \neg x \land \neg y) \\
&\equiv \big((a \to b) \land a\big) \land \neg b && (\neg\neg x \equiv x) \\
&\equiv (\neg a \lor b) \land a \land \neg b && (x \to y \equiv \neg x \lor y) \\
&\equiv (\neg a \lor b) \land a \land \neg b && ((x \lor y) \land z \equiv (x \land z) \lor (y \land z)) \\
&\equiv (\neg a \land a \land \neg b) \lor (b \land a \land \neg b) &&
\end{aligned}$$

Contradicción

Ya tenemos la fórmula en DNF, ahora tenemos que demostrar la contradicción. Lo hacemos refutando cada cláusula

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

Reglas

$$\frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}\ \mathrm{E}\vee$$

$$\frac{\Gamma \vdash \neg A \qquad \Gamma \vdash A}{\Gamma \vdash \bot}\ \mathrm{E}\neg$$

Recap by

Teniendo en el contexto $\Gamma = \{h_1 : b_1, \ldots, h_n : b_n\}$ para certificar

$$\text{thus } a \text{ by } h_1 \ldots h_n$$

- Debe demostrar $b_1 \land \ldots \land b_n \to a$
- Lo hace por absurdo: la niega y encuentra una contradicción
- Primero la convierte a forma normal disyuntiva (DNF)
- Luego refuta cada cláusula (conjunción de literales)
  - False $(\bot \land p \land q)$
  - Literales opuestos $(p(a) \land \neg p(a) \land q)$
  - Eliminación de existencial $(\forall x. p(x) \land \neg p(a))$

Desafío: ¡Hay que generar una demostración de deducción natural!

DNF

¿Cómo demostramos el pasaje de uno al otro?

$$\neg[((a \rightarrow b) \wedge a) \rightarrow b] \vdash \bot$$

$$\vdots$$

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

Generando demostraciones para todas las equivalencias, y convirtiendo la fórmula paso por paso (*"small step"*)

$$\begin{aligned} \neg\neg x &\equiv x \\ \neg\bot &\equiv \top \\ \neg\top &\equiv \bot \\ x \rightarrow y &\equiv \neg x \vee y \\ \neg(x \vee y) &\equiv \neg x \wedge \neg y \\ \neg(x \wedge y) &\equiv \neg x \vee \neg y \end{aligned}$$

$$\begin{aligned} (x \vee y) \wedge z &\equiv (x \wedge z) \vee (y \wedge z) \\ z \wedge (x \vee y) &\equiv (z \wedge x) \vee (z \wedge y) \\ x \vee (y \vee z) &\equiv (x \vee y) \vee z \\ x \wedge (y \wedge z) &\equiv (x \wedge y) \wedge z \end{aligned}$$

DNF

Para poder hacerlo paso por paso también hace falta demostrar la *congruencia* de los operadores

$$a \vee \neg(b \vee c) \equiv a \vee (\neg b \wedge \neg c)$$

En general,

$$\alpha \equiv \alpha' \Rightarrow \alpha \wedge \beta \equiv \alpha' \wedge \beta$$
$$\beta \equiv \beta' \Rightarrow \alpha \wedge \beta \equiv \alpha \wedge \beta'$$

Análogo para $\vee$, $\neg$

¿Por qué este mecanismo?

- Es un procedimiento completo para LP pero **heurístico** para LPO, puede fallar (i.e no demuestra cualquier cosa)
- Satisfacibilidad de LPO es indecidible
- Mecanismos como *resolución general* se pueden colgar
- Podríamos haber hecho otro, queríamos hacer *alguno*

Friedman

Lógica clásica / Lógica intuicionista

- La lógica **clásica** no siempre es constructiva, por el *principio del tercero excluido* (LEM):

  para toda proposición A, es verdadera ella o su negación
  $$A \vee \neg A$$

- La lógica **intuicionista** se puede describir de forma sucinta como la lógica clásica sin LEM. Equivalentemente, tampoco vale la *eliminación de la doble negación* ($\neg\neg A \to A$)

Demostración no constructiva

Teorema

Existen dos números irracionales a, b tq a^b es racional.

Sabemos que $\sqrt{2}$ es irracional, y por LEM que $\sqrt{2}^{\sqrt{2}}$ es o racional o irracional.

- Si $\sqrt{2}^{\sqrt{2}}$ es racional, tomamos $a = b = \sqrt{2}$
- Sino, tomamos $a = \sqrt{2}^{\sqrt{2}}$, $b = \sqrt{2}$ y luego

$$a^b = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\sqrt{2}} = \sqrt{2}^2 = 2$$

que es racional.

¡No nos dice cuales son a y b!

Traducción de Friedman

- Queremos "reducir" o "ejecutar" los programas para obtener testigos de existenciales.
- La lógica clásica no es ejecutable (no constructiva). La intuicionista sí
- Friedman traduce de clásica a intuicionista. Caveat: solo fórmulas $\in \Pi_2^0$ (i.e de la forma $\forall x_1 \ldots \forall x_n \exists y . \varphi$)

¿En dónde estamos parados?