Fundamentos formales de especificaciones algebraicas heterogéneas

Carlos G. Lopez Pombo
Departamento de computación, FCEyN, UBA
y CONICET

Especificaciones

En un sentido clásico, especificar permite documentar **qué** hace un sistema o pieza de software.

Fundamentos formales

Los fundamentos formales aportan la posibilidad de que una descripción (especificación) carezca de ambigüedades y, por tanto, posea una única lectura posible.

Algebraicas

En este punto la cosa se vuelve más subjetiva. Algunas propiedades interesantes a favor de las álgebras:

- expresan matemáticamente cómo se obtienen ciertos "objetos" a partir de otros por medio de la aplicación de "operaciones",
- son la semántica por defecto de gran cantidad de lógicas, un lenguaje apropiado para alcanzar especificaciones rigurosas y formales,
- han sido estudiadas por mucho tiempo y consecuentemente se cuenta con gran cantidad de herramientas para comprenderlas en profundidad.

Heterogéneas

Los sistemas de software son naturalmente heterogéneos. Un mismo sistema posee características diversas para las que no existe un único lenguaje apropiado para especificarlas.

Temario

- Lógica, TADs y teoría de categorías para principiantes
- Caracterización categórica de una lógica (Instituciones)
- Composicionalidad y propiedades emergentes
- Especificaciones heterogéneas
- Especificaciones estructuradas

Lógica, TADs y teoría de categorías para principiantes

Carlos G. Lopez Pombo
Departamento de computación, FCEyN, UBA
y CONICET

Lógica

[End72]

¿Qué es “una lógica”?

Lógica

[End72]

¿Qué es "una lógica"?

- **Sintaxis:** una descripción del conjunto de fórmulas bien formadas (usualmente se provee a través de una gramática)
- **Semántica:** una clase de estructuras que dan significado a una fórmula
- **Satisfactibilidad:** una manera de saber si una estructura dada satisface una fórmula.

Lógica de primer orden

(sintaxis)

Términos

Dado $\{\mathrm{P}_i\}_{i\in\mathcal{I}}$ un conjunto de símbolos de relación, $\{\mathrm{f}_j\}_{j\in\mathcal{J}}$ un conjunto de símbolos de función y $\{v_k\}_{k\in\mathcal{K}}$ un conjunto de símbolos de variables, se denota $TermFOL$ al menor conjunto T tal que:

- $\{v_k\}_{k\in\mathcal{K}} \subseteq T$
- Si $t_1, \ldots, t_n \in T$ y $arity(f_i) = n$, entonces $f_i(t_1, \ldots, t_n) \in T$

Lógica de primer orden

(sintaxis)

Fórmulas

Dado $\{\mathrm{P}_i\}_{i\in\mathcal{I}}$ un conjunto de símbolos de relación, $\{\mathrm{f}_j\}_{j\in\mathcal{J}}$ un conjunto de símbolos de función y $\{v_k\}_{k\in\mathcal{K}}$ un conjunto de símbolos de variables, se denota $FormFOL$ al menor conjunto F tal que:

- Si $t_1, t_2 \in TermFOL$, entonces $t_1 = t_2 \in F$
- Si $t_1, \ldots, t_n \in TermFOL$ y $arity(P_i) = n$, entonces $P_i(t_1, \ldots, t_n) \in F$
- Si $\alpha, \beta \in F$ y $v \in \{v_k\}_{k\in\mathcal{K}}$, entonces $\neg\alpha, \alpha \vee \beta, (\exists v)\alpha \in F$

Lógica de primer orden

(sintaxis)

Dada $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura, se denota Sen_Σ al conjunto de formulas bien formadas sin variables libres sobre Σ.

Lógica de primer orden

(semántica)

Dado $\{\mathrm{P}_i\}_{i\in\mathcal{I}}$ un conjunto de símbolos de relación, $\{\mathrm{f}_j\}_{j\in\mathcal{J}}$ un conjunto de símbolos de función, una estructura $\mathcal{M}$ para interpretar $\langle\{P_i\}_{i\in\mathcal{I}}, \{f_j\}_{j\in\mathcal{J}}\rangle$ (una signatura) es:

$\langle M, \{P_i^{\mathcal{M}}\}_{i\in\mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j\in\mathcal{J}}\rangle$ **tal que:**

- **Para todo** $i \in \mathcal{I}$**,** $P_i^{\mathcal{M}} \subseteq M^{arity(P_i)}$
- **Para todo** $j \in \mathcal{J}$**, si** $arity(f_j) = n$ **y** $e_1, \ldots, e_n \in M$**,**

entonces $f_j^{\mathcal{M}}(e_1, \ldots, e_n) \in M$

Lógica de primer orden

(semántica)

Dada $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura, se denota Mod_Σ a la clase de estructuras sobre las que se puede interpretar a Σ.

Lógica de primer orden

(satisfactibilidad)

Valuaciones

Sea $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura, $\mathcal{M} = \langle M, \{P_i^{\mathcal{M}}\}_{i \in \mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j \in \mathcal{J}} \rangle$ una estructura, una valuación de las variables es $v : \{v_k\}_{k \in \mathcal{K}} \to M$.

$$v[y \to e](x) = \begin{cases} e & , \text{si } x = y. \\ v(x) & , \text{si } x \neq y. \end{cases}$$

Lógica de primer orden

(satisfactibilidad)

Términos

Sea $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura,
$\mathcal{M} = \langle M, \{P_i^{\mathcal{M}}\}_{i \in \mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j \in \mathcal{J}} \rangle$ una estructura y
$v : \{v_k\}_{k \in \mathcal{K}} \to M$ una valuación de las variables en M
se define $m^{\mathcal{M}} : TermFOL \to M$ de la siguiente forma:

- Si $x \in \{v_k\}_{k \in \mathcal{K}}$, entonces $m_v^{\mathcal{M}}(x) = v(x)$
- Si $t_1, \ldots, t_n \in TermFOL$ y $arity(f_j) = n$, entonces

$$m_v^{\mathcal{M}}(f_j(t_1, \ldots, t_n)) = f_j^{\mathcal{M}}(m_v^{\mathcal{M}}(t_1), \ldots, m_v^{\mathcal{M}}(t_n))$$

Lógica de primer orden

(satisfactibilidad)

Fórmulas

Sea $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura,
$\mathcal{M} = \langle M, \{P_i^{\mathcal{M}}\}_{i \in \mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j \in \mathcal{J}} \rangle$ una estructura y
$v : \{v_k\}_{k \in \mathcal{K}} \to M$ una valuación de las variables en M
se define $\models_v^{\Sigma} \subseteq Mod_{\Sigma} \times Sen_{\Sigma}$ de la siguiente forma:

$$
\begin{array}{rcl}
\mathcal{M} \models_v^{\Sigma} t1 = t2 & \text{sii} & m_v^{\mathcal{M}}(t_1) = m_v^{\mathcal{M}}(t_2) \\
\mathcal{M} \models_v^{\Sigma} P_i(t_1, \ldots, t_n) & \text{sii} & (m_v^{\mathcal{M}}(t_1), \ldots, m_v^{\mathcal{M}}(t_n)) \in P_i^{\mathcal{M}} \\
\mathcal{M} \models_v^{\Sigma} \neg\alpha & \text{sii} & \text{no pasa que } \mathcal{M} \models_v^{\Sigma} \alpha \\
\mathcal{M} \models_v^{\Sigma} \alpha \vee \beta & \text{sii} & \mathcal{M} \models_v^{\Sigma} \alpha \text{ o } \mathcal{M} \models_v^{\Sigma} \beta \\
\mathcal{M} \models_v^{\Sigma} (\exists x)\alpha & \text{sii} & \text{existe } e \in M \text{ tal que } \mathcal{M} \models_{v[x \to e]}^{\Sigma} \alpha
\end{array}
$$

Lógica de primer orden

(paréntesis)

¿No hay nada que les llame la atención de las dos definiciones anteriores (y de la de signatura)?

Lógica de primer orden

(paréntesis)

¿No hay nada que les llame la atención de las dos definiciones anteriores (y de la de signatura)?

Los símbolos con los que se pueden construir fórmulas se dividen en dos clases...

- **Los lógicos**: tienen una única interpretación (semántica) y se mantiene en todas las estructuras de la clase Mod_Σ. $(\neg, \vee, \exists)$
- **Los extralógicos:** la interpretación depende de la interpretación m_v^Σ que se tome para cada estructura de la clase Mod_Σ. $(\{P_i\}_{i\in I}, \{f_j\}_{j\in J})$

Lógica de primer orden

Sea $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura,
$\mathcal{M} = \langle M, \{P_i^{\mathcal{M}}\}_{i \in \mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j \in \mathcal{J}} \rangle$ una estructura.

$$\mathcal{M} \models^{\Sigma} \Gamma \quad \text{sii} \quad \text{para toda } \alpha \in \Gamma,\ \mathcal{M} \models^{\Sigma} \alpha$$
$$\Gamma \models^{\Sigma} \alpha \quad \text{sii} \quad \text{para toda } \mathcal{M} \in Mod_{\Sigma},\ \text{si } \mathcal{M} \models^{\Sigma} \Gamma,\ \text{entonces } \mathcal{M} \models^{\Sigma} \alpha$$

$$\alpha \text{ es satisfacible} \quad \text{sii} \quad \text{existe } \mathcal{M} \in Mod_{\Sigma},\ \text{tal que } \mathcal{M} \models^{\Sigma} \alpha$$
$$\alpha \text{ es válida} \quad \text{sii} \quad \text{para toda } \mathcal{M} \in Mod_{\Sigma},\ \text{tal que } \mathcal{M} \models^{\Sigma} \alpha$$

Nota: $\models^{\Sigma}$ denota la relación de satisfactibilidad $\models^{\Sigma}_{\emptyset}$.

Lógica de primer orden

Sea $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura,
$\mathcal{M} = \langle M, \{P_i^{\mathcal{M}}\}_{i \in \mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j \in \mathcal{J}} \rangle$ una estructura.

Dada $\Gamma \subseteq Sen_\Sigma$, $\Gamma^\bullet$ satisface $\Gamma \models^\Sigma \alpha$ sii $\alpha \in \Gamma^\bullet$

Dada $\Gamma \subseteq Sen_\Sigma$, $\langle \Sigma, \Gamma^\bullet \rangle$ se denomina teoría y $\langle \Sigma, \Gamma \rangle$ es una presentación de la teoría $\langle \Sigma, \Gamma^\bullet \rangle$.

Dada $\Gamma \subseteq Sen_\Sigma$,

$$Mod_{\langle \Sigma, \Gamma \rangle} = \{\mathcal{M} | \mathcal{M} \in Mod_\Sigma \text{ y } \mathcal{M} \models^\Sigma \Gamma\}$$

Lógica de primer orden

Sea $\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$ una signatura,
$\mathcal{M} = \langle M, \{P_i^{\mathcal{M}}\}_{i \in \mathcal{I}}, \{f_j^{\mathcal{M}}\}_{j \in \mathcal{J}} \rangle$ una estructura.

Lema: Dada $\Gamma \subseteq Sen_\Sigma$,

$$Mod_{\langle \Sigma, \Gamma \rangle} = Mod_{\langle \Sigma, \Gamma \bullet \rangle}$$

Demostración: ...

Tipos abstractos de dato

Normalmente se denomina TAD a una descripción formal de ciertos objetos del mundo real a partir de comportamiento de las operaciones que aplican sobre ellos.

Tipos abstractos de dato

TAD Bool
genero: bool
usa:
exporta: bool, true, false, not, or
generadores:
true: $\longrightarrow$ bool
false: $\longrightarrow$ bool
observadores:

otras operaciones:
not: bool $\longrightarrow$ bool
or: bool x bool $\longrightarrow$ bool
axiomas:
not (true) = false not (false) = true
x or true = true x or false = x
Fin TAD

Tipos abstractos de dato

TAD Nat
genero: nat
usa: bool
exporta: nat, 0, suc, +, *
generadores:

0: $\longrightarrow$ nat

suc: nat $\longrightarrow$ nat

observadores:

es0?: nat $\longrightarrow$ bool

pred: x: nat $\longrightarrow$ nat not es0? (x)

otras operaciones:

+: nat x nat $\longrightarrow$ nat

*: nat x nat $\longrightarrow$ nat

axiomas:

es0? (0) = true es0? (suc (x)) = false

pred (suc (x)) = x

x + 0 = x x + suc (y) = suc (x + y)

x * 0 = 0 x * suc (y) = (x * y) + x

Fin TAD

TADs vs. lógica

¿En qué se parecen los TADs con la lógica formal tal como la presentamos antes?

TADs vs. lógica

¿En qué se parecen los TADs con la lógica formal tal como la presentamos antes?

¡En todo! Son la misma cosa pero presentada en forma diferente

TADs vs. lógica

TAD Bool
genero: bool
usa:
exporta: bool, true, false, not, or
generadores:

true: $\longrightarrow$ bool

false: $\longrightarrow$ bool

observadores:

otras operaciones:

not: bool $\longrightarrow$ bool

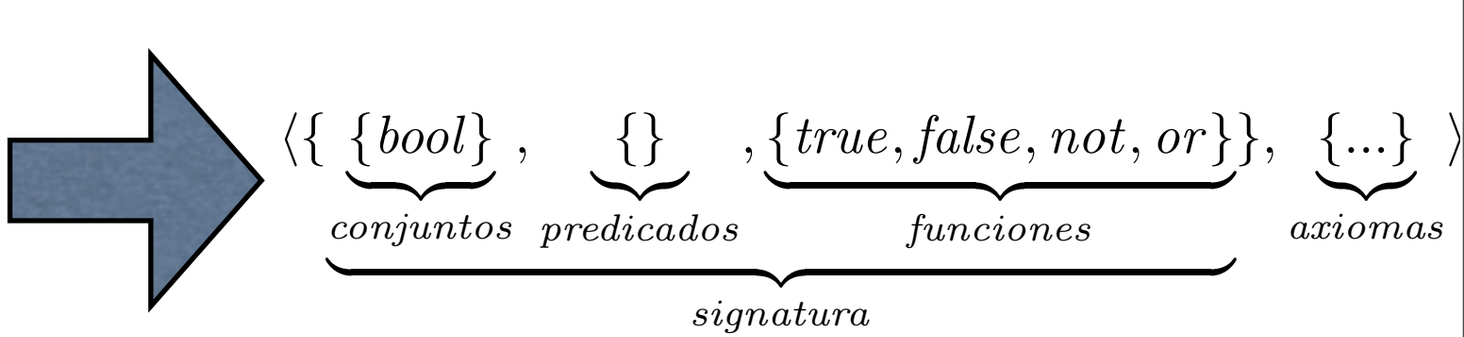
or: bool x bool $\longrightarrow$ bool

axiomas:

not (true) = false not (false) = true

x or true = true x or false = x

Fin TAD



TADs vs. lógica

TAD Nat
genero: nat
usa: bool
exporta: nat, 0, suc, +, *
generadores:
0: $\longrightarrow$ nat
suc: nat $\longrightarrow$ nat
observadores:
es0?: nat $\longrightarrow$ bool
pred: x: nat $\longrightarrow$ nat
otras operaciones:
+: nat x nat $\longrightarrow$ nat
*: nat x nat $\longrightarrow$ nat
axiomas:
es0? (0) = true
es0? (suc (x)) = false
pred (suc (x)) = x
x + 0 = x
x + suc (y) = suc (x + y)
x * 0 = 0
x * suc (y) = (x * y) + x
Fin TAD

not es0? (x)

$$\langle\{\underbrace{\{nat, bool\}}_{conjuntos}, \underbrace{\{es0?\}}_{predicados}, \underbrace{\{0, suc, pred, +, *, false, true\}}_{funciones}\}, \{\underbrace{\ldots}_{axiomas}\}\rangle$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{signatura}$$

Modularidad

Una **enorme ventaja** de lenguajes estructurados como el que acabamos de ver para los TADs es que resultan ser modulares.

Esto quiere decir que no tenemos necesidad de construir toda la especificación en forma **monolítica**, sino que podemos hacerlo **por partes** para luego juntar todo en una única descripción formal del sistema.

Problemas...

$$\langle \{ \underbrace{\{bool\}}_{conjuntos}, \underbrace{\{\}}_{predicados}, \underbrace{\{true, false, not, or\}}_{funciones} \}, \underbrace{\{...\}}_{axiomas} \rangle$$
$$signatura$$

$$\langle \{ \underbrace{\{nat, bool\}}_{conjuntos}, \underbrace{\{es0?\}}_{predicados}, \underbrace{\{0, suc, pred, +, *, false, true\}}_{funciones} \}, \{ \underbrace{...}_{axiomas} \} \rangle$$
$$signatura$$

- ¿Cómo hago para pegar las partes en el todo?
- ¿Qué tienen que ver true, false y bool de la primera teoría con true, false y bool de la segunda?¿Cómo hago para vincular símbolos?

Problemas...

$$\langle \{ \underbrace{\{bool\}}_{conjuntos}, \underbrace{\{\}}_{predicados}, \underbrace{\{true, false, not, or\}}_{funciones} \}, \underbrace{\{...\}}_{axiomas} \rangle$$
$$signatura$$

$$\langle \{ \underbrace{\{nat, bool\}}_{conjuntos}, \underbrace{\{es0?\}}_{predicados}, \underbrace{\{0, suc, pred, +, *, false, true\}}_{funciones} \}, \{ \underbrace{...}_{axiomas} \} \rangle$$
$$signatura$$

- ¿Cómo hago para pegar las partes en el todo?
- ¿Qué tienen que ver true, false y bool de la primera teoría con true, false y bool de la segunda?¿Cómo hago para vincular símbolos?

¡Ajá!... ¿y si no tuvieran el mismo nombre pero representaran el mismo objeto? Recordemos que una cosa buena de la modularidad es que las especificaciones son independientes.

Soluciones...

Nuestra primera misión, para mañana, será estudiar la forma en la que podemos resolver este problema nada menor. Si no resolvemos cómo **componer** las partes en el todo, deberemos olvidarnos de la modularidad... cosa que no pensamos hacer **:-)**

Ahora vamos a estudiar las herramientas básicas que necesitaremos para resolver este problema.

“Teoría de categorías” [McL71, Fia05]

A diferencia de la teoría de conjuntos, que caracteriza a los objetos por sus propiedades, la teoría de categorías caracteriza a los objetos por sus relaciones entre sí... en palabras de Jean-Yves Girard, por sus “relaciones sociales”.

“Teoría de categorías” [McL71, Fia05]

A diferencia de la teoría de conjuntos, que caracteriza a los objetos por sus propiedades, la teoría de categorías caracteriza a los objetos por sus relaciones entre sí... en palabras de Jean-Yves Girard, por sus “relaciones sociales”.

Esto es lo que necesitaremos para poder relacionar signaturas diferentes... y por lo tanto “pegar” sus símbolos para componerlas razonablemente.

"Teoría de categorías"

Grafo:

[Fia05]

A *graph* is a structure $\langle G_0, G_1 \rangle$ where

- G_0 is a collection (of nodes),
- G_1 is a collection (of arrows),
- $src : G_1 \to G_0$ maps every arrow to a node (the source of the arrow),
- $trg : G_1 \to G_0$ maps every arrow to a node (the target of the arrow).

If $f \in G_1$ is an arrow from x to y, it will be denoted as $f : x \to y$ or $x \xrightarrow{f} y$.

“Teoría de categorías”

Homomorfismo de un grafo: [Fia05]

Let G, and H be graphs. A *homomorphism* of graphs $\phi : G \to H$ is a pair of maps $\phi_0 : G_0 \to H_0$ and $\phi_1 : G_1 \to H_1$ such that for each arrow $f : x \to y \in G_1$, we have $\phi_1(f) : \phi_0(x) \to \phi_0(y) \in H_1$.

“Teoría de categorías”

Camino en un grafo: [Fia05]

Let $G = \langle G_0, G_1 \rangle$ be a graph, and $x, y \in G_0$. A *path* from x to y of length $k > 0$ is a sequence $f_1, \ldots, f_k$ such that:

- $f_i \in G_1$ $(1 \leq i \leq k)$,
- $src(f_1) = x$,
- $trg(f_i) = src(f_{i+1})$ $(1 < i < k)$,
- $trg(f_k) = y$.

We will denote by G_i the collection of paths of length i.

"Teoría de categorías"

Categoría: [Fia05]

A *category* is a structure $\langle G, \circ, id \rangle$ where:

- $G = \langle G_0, G_1 \rangle$ is a graph,
  - $\circ : G_2 \to G_1$ is a map from G_2 into G_1 (called composition). If $f, g \in G_1$, then the composition of f and g is denoted as $f \circ g : x \to z$, and
  - $id : G_0 \to G_1$ is a map from G_0 into G_1 (called identity). If $x \in G_0$, $id_x : x \to x$ is the identity arrow for x,

such that for all $x, y \in G_0$ and $f, g, h \in G_1$:

- if $f : x \to y \in G_0$, then $f = id_x \circ f = f \circ id_y$,
- if $src(g) = trg(f)$ and $src(h) = trg(g)$, then $(f \circ g) \circ h = f \circ (g \circ h)$.

If C is a category, by $graph(\mathsf{C})$ we denote the graph of C.

Elements in a category are called *objects* and arrows are referred to as *morphisms*.

“Teoría de categorías”

Categoría (Set): [McL71]

- Objetos: la clase de todos los conjuntos,
- Morfismos: las funciones totales entre conjuntos.

Lema: Set es una categoría.

Demostración: ...

"Teoría de categorías"

Categoría (Set): [McL71]

- **Teoría de conjuntos:** los objetos (conjuntos) se caracterizan por los elementos que contienen.

$$C = C' \quad \text{sii} \quad \text{para todo } e, e \in C \text{ sii } e \in C'$$

- **Teoría de categorías:** los objetos (conjuntos) se caracterizan por las funciones totales en las que participan.

$$C = C' \quad \text{sii} \quad \text{para todo } e, hom(e, C) = hom(e, C') \text{ y}$$
$$\text{para todo } e, hom(C, e) = hom(C', e)$$

“Teoría de categorías”

Categoría (Set): ¿Quién es ∅? **[McL71]**

- **Teoría de conjuntos:** es el único conjunto C tal que para todo elemento e, e no pertenece a C.

- **Teoría de categorías:** es el único objeto tal que existe un único morfismo desde él a cualquier otro objeto de la categoría.

Caracterización categórica de una lógica (Instituciones)

Carlos G. Lopez Pombo
Departamento de computación, FCEyN, UBA
y CONICET

La categoría de las signaturas de la lógica de primer orden

Let $Sign$ be the class of all possible first-order logic signatures. Then, we define $\mathsf{Sign} = \langle Sign, \mathcal{A} \rangle$, where

$$\mathcal{A} = \left\{ \left\langle \ \sigma_P : \{P_i\}_{i \in \mathcal{I}} \to \{P'_i\}_{i \in \mathcal{I}'}, \sigma_f : \{f_j\}_{j \in \mathcal{J}} \to \{f'_j\}_{j \in \mathcal{J}'} \ \right\rangle \mid \begin{array}{c} \sigma_1 : \mathcal{I} \to \mathcal{I}' \text{ is a total function, such that} \\ (\forall i \in \mathcal{I})(arity(P_i) = arity(P'_{\sigma_1(i)})) \ . \\ \sigma_2 : \mathcal{J} \to \mathcal{J}' \text{ is a total function, such that} \\ (\forall j \in \mathcal{J})(arity(f_j) = arity(f'_{\sigma_2(j)})) \ . \end{array} \right\}$$

If $\Sigma \in Sign$, then $id_\Sigma = \langle id : \mathcal{I} \to \mathcal{I}, id : \mathcal{J} \to \mathcal{J} \rangle$.

If $f = \langle f_1, f_2 \rangle : \Sigma \to \Sigma'$ y $g = \langle g_1, g_2 \rangle : \Sigma' \to \Sigma''$, then $f \circ g = \langle f_1 \circ g_1, f_2 \circ g_2 \rangle$.

Lema: Sign es una categoría.

Demostración: ...

Nuestro objetivo

Hemos visto cómo un concepto que en principio era individual como el de signatura, se ha convertido en algo mucho más complejo (e interesante) que es la categoría de las signaturas.

Nuestro objetivo

Hemos visto cómo un concepto que en principio era individual como el de signatura, se ha convertido en algo mucho más complejo (e interesante) que es la categoría de las signaturas.

Nuestra misión a partir de ahora será formalizar el concepto de lógica en teoría de categorías.

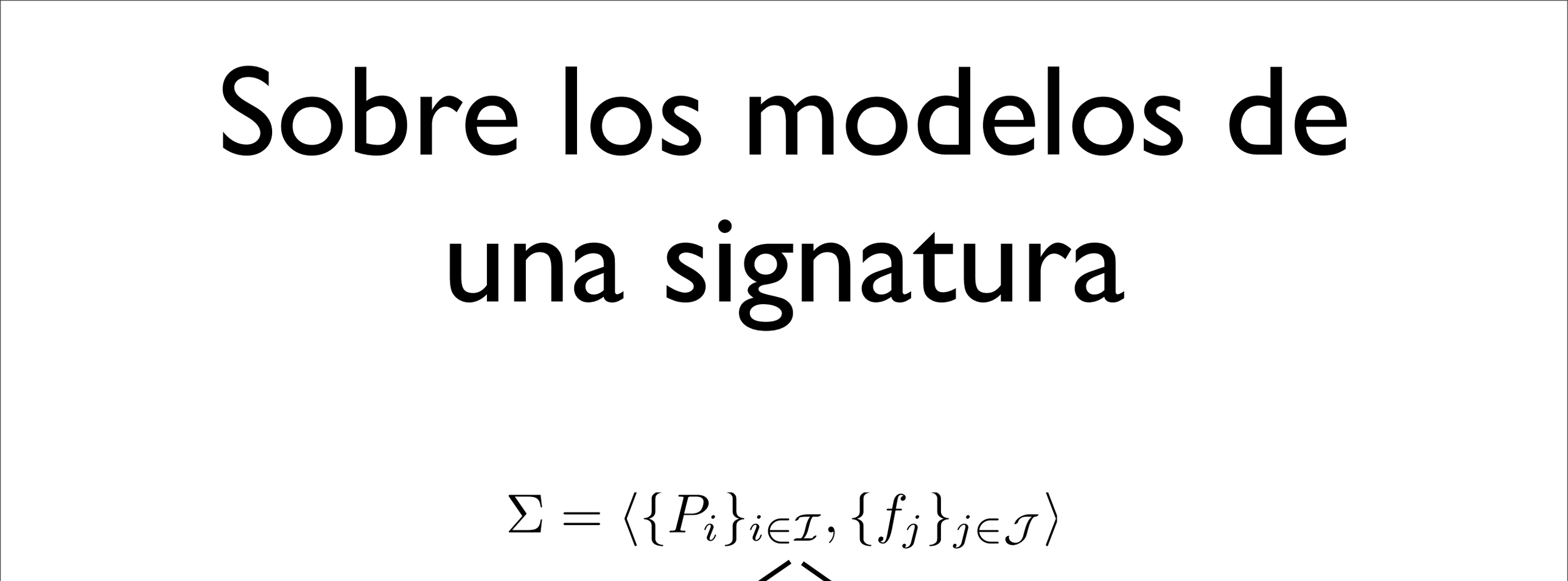
Sobre los modelos de una signatura

Dada una signatura Σ ya hemos discutido sobre Mod_Σ...

Si todas las estructuras de Mod_Σ son interpretaciones de Σ. ¿Será que existe entre ellas relaciones que la conviertan en una categoría?

Sobre los modelos de una signatura

$$\Sigma = \langle \{P_i\}_{i \in \mathcal{I}}, \{f_j\}_{j \in \mathcal{J}} \rangle$$



Sobre los modelos de una signatura

Tipo de un álgebra

[BS81]

Let $\mathcal{T} = \langle \mathcal{A}, \mathcal{F} \rangle$ be a language. Then an *algebra* $\mathfrak{A}$ of similarity type $\mathcal{T}$ is a structure $\langle \mathcal{A}, \{(f : A_{i_1} \times \ldots \times A_{i_n} \to A_i)_i\}_{i \in \mathcal{I} \wedge \{A_{i_1}, \ldots, A_{i_n}, A_i\} \subseteq \mathcal{A}} \rangle$ such that:

$$f_i : A_{i_1} \times \ldots \times A_{i_n} \to A_i \text{ for all } i \in \mathcal{I}.$$

Nota: Por comodidad en lo que sigue trabajaremos en la lógica ecuacional, y por lo tanto no habrá símbolos de predicado.

Sobre los modelos de una signatura

Homomorphismo

[BS81]

Let $\mathfrak{A} = \langle A, \{f_i\}_{i \in \mathcal{I}} \rangle$ and $\mathfrak{B} = \langle B, \{g_i\}_{i \in \mathcal{I}} \rangle$ two algebras of the same similarity type, a function $h : A \to B$ is a *homomorphism* from $\mathfrak{A}$ to $\mathfrak{B}$ if

$$h(f_i(a_1, \ldots, a_{arity(f_i)})) = g_i(h(a_1), \ldots, h(a_{arity(g_i)})) \text{ for all } i \in \mathcal{I}.$$

La categoría de los modelos de una signatura

Let $\Sigma \in |\mathsf{Sign}|$, then $\mathsf{Mod}_\Sigma = \langle \mathcal{O}, \mathcal{A} \rangle$ is defined as follows:

- $\mathcal{O} = Mod_\Sigma$,
- $\mathcal{A} = \{\gamma : \mathcal{M} \rightarrow \mathcal{M}' \mid \mathcal{M}, \mathcal{M}' \in \mathcal{O} \text{ y } \gamma \text{ es un homomorfismo}\}$.

Lema: Mod_Σ **es una categoría.**

Demostración: ...

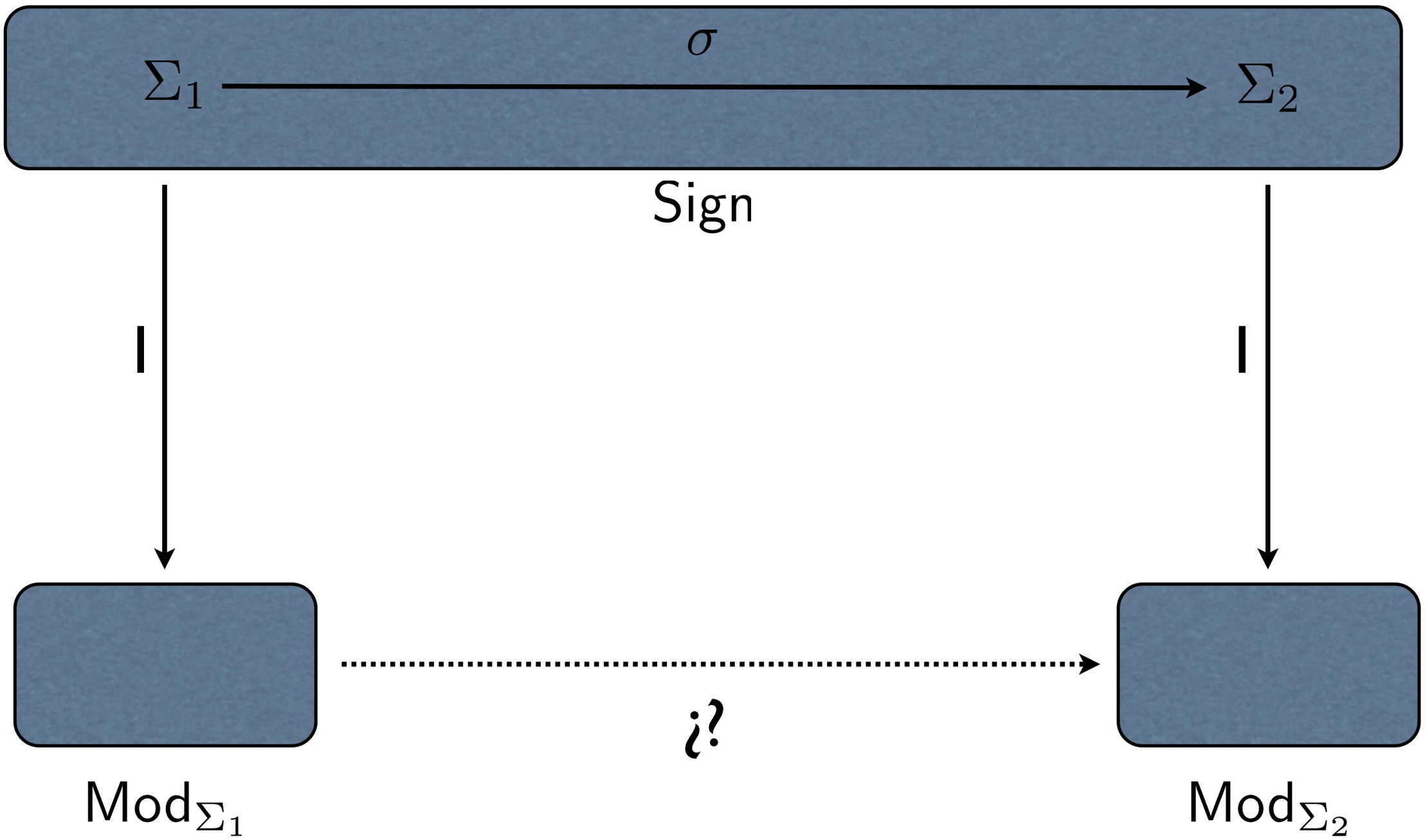
En resumen...

- La clase de las signaturas (sean ecuacionales o de primer orden) con los morfismos como fueron definidos forman una categoría,
- Dada una signatura, su clase de modelos con los homomorphismos forman una categoría,

Sobre los modelos de diferentes signaturas

Si existen relaciones entre las signaturas (morfismos en Sign), y a cada una de ellas le corresponde una categoría de modelos; dadas dos signaturas Σ_1 y Σ_2 en Sign, ¿hay alguna relación entre Mod_{Σ_1} y Mod_{Σ_2}?

Sobre los modelos de diferentes signaturas



De las relaciones entre categorías

Functor

Let $\mathsf{C} = \langle \mathcal{O}_\mathsf{C}, \mathcal{A}_\mathsf{C} \rangle$ and $\mathsf{D} = \langle \mathcal{O}_\mathsf{D}, \mathcal{A}_\mathsf{D} \rangle$ be categories. Then $\delta : \mathsf{C} \to \mathsf{D}$ is a *functor* if and only if:

- if $x \in \mathcal{O}_\mathsf{C}$, then $\delta(x) \in \mathcal{O}_\mathsf{D}$,
- if $f : x \to y \in \mathcal{A}_\mathsf{C}$, then $\delta(f) \in \mathcal{A}_\mathsf{D}$, such that:
  - if $x \in \mathcal{O}_\mathsf{C}$, $\delta(id^\mathsf{C}{}_x) = id^\mathsf{D}{}_{\delta(x)}$,
  - if $f : x \to y, g : y \to z \in \mathcal{A}_\mathsf{C}$, then $\delta(f \circ_\mathsf{C} g) = \delta(f) \circ_\mathsf{D} \delta(g)$.

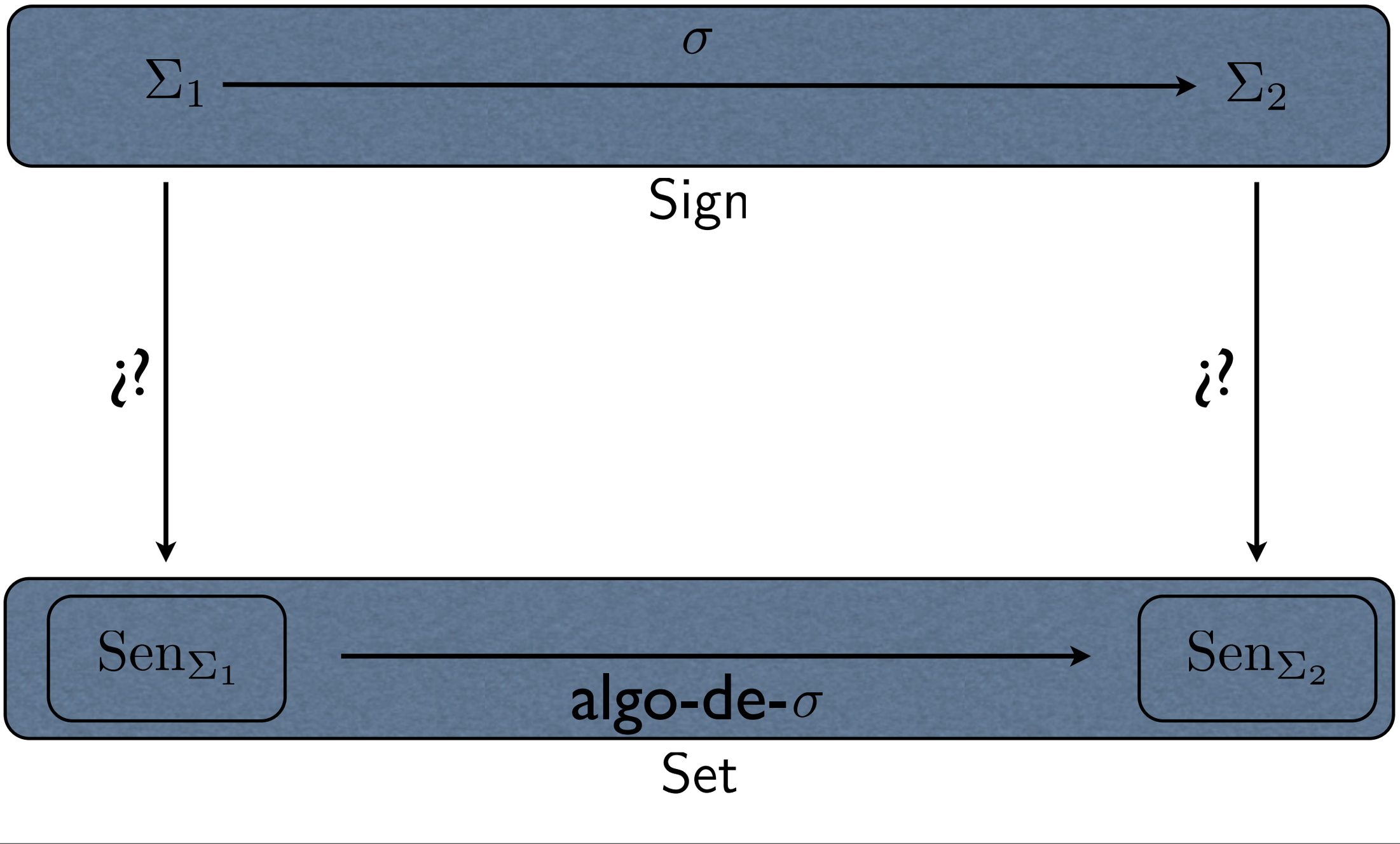
De las relaciones entre categorías

Functor

[Fia05]

Los functores son una correspondencia entre los objetos de dos categorías de forma que las relaciones entre dos objetos de la primera son preservadas en sus imágenes en la segunda.

Un ejemplo de functor



Un ejemplo de functor

¿Qué debiera hacer nuestro functor?
Debiera caracterizar la función algo-de-σ en términos del comportamiento de σ.

- A cada sentencia de Sen_{Σ_1} debe corresponderle una de Sen_{Σ_2}, y
- Debe preservar las identidades y composiciones.

Un ejemplo de functor

$$\Sigma_1 = \langle \{P_i^1\}_{i\in\mathcal{I}_1}, \{f_j^1\}_{j\in\mathcal{J}_1}\rangle \xrightarrow{\sigma = \langle \sigma_P, \sigma_f\rangle} \Sigma_2 = \langle \{P_i^2\}_{i\in\mathcal{I}_2}, \{f_j^2\}_{j\in\mathcal{J}_2}\rangle$$

$$\begin{aligned}
T(\sigma)(v_k) &= v_k \\
T(\sigma)(f_j^1(t_1,\ldots,t_n)) &= f^2_{\sigma_f(j)}(T(\sigma)(t_1),\ldots,T(\sigma)(t_n)) \\
T(\sigma)(t_1 = t_2) &= T(\sigma)(t_1) = T(\sigma)(t_2) \\
T(\sigma)(P_i^1(t_1,\ldots,t_n)) &= P^2_{\sigma_P(i)}(T(\sigma)(t_1),\ldots,T(\sigma)(t_n)) \\
T(\sigma)(\neg\alpha) &= \neg T(\sigma)(\alpha) \\
T(\sigma)(\alpha\vee\beta) &= T(\sigma)(\alpha)\vee T(\sigma)(\beta) \\
T(\sigma)((\exists v)\alpha) &= (\exists v)T(\sigma)(\alpha)
\end{aligned}$$

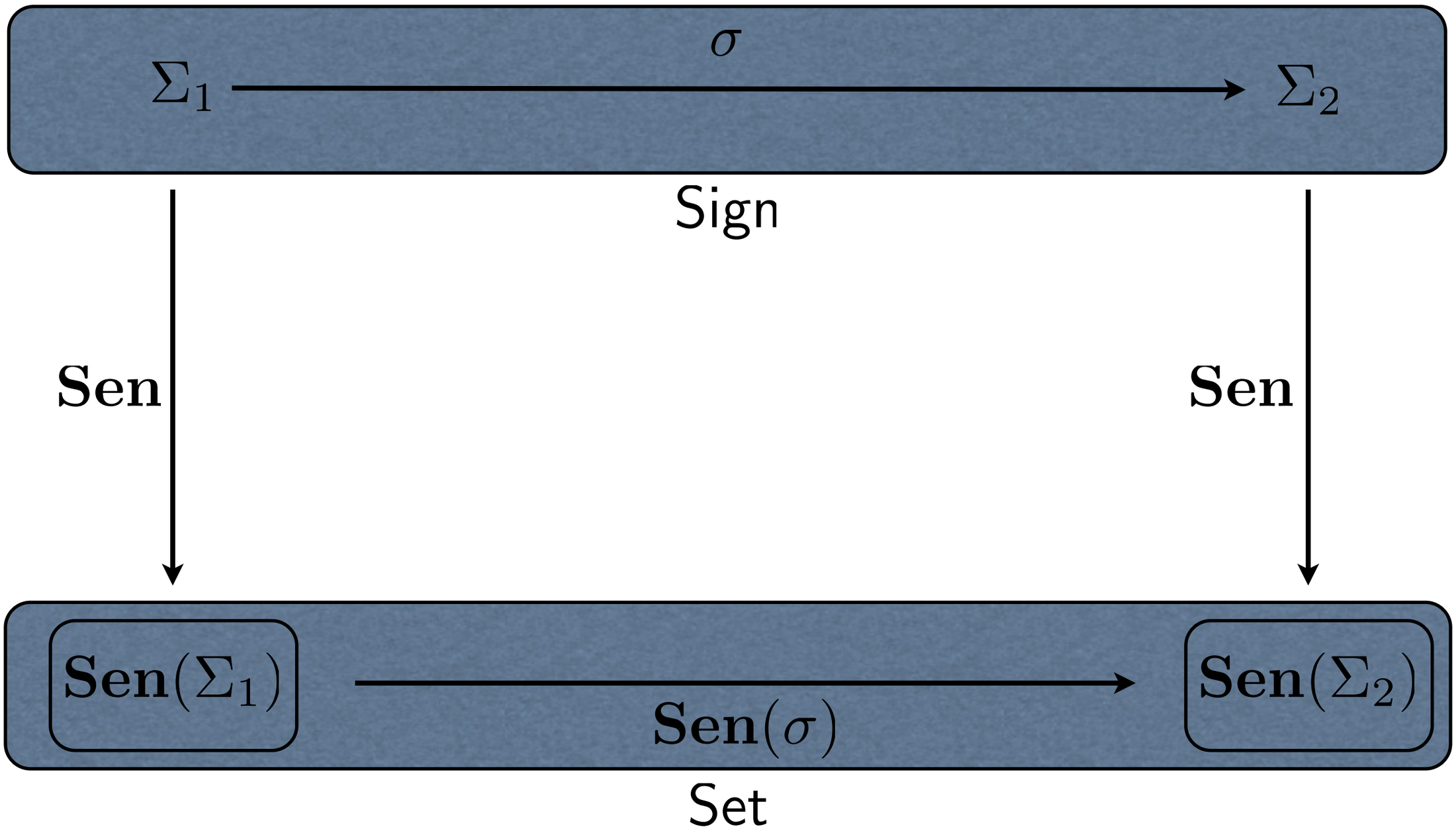
Un ejemplo de functor

Si definimos T de forma que a cada signatura Σ le haga corresponder Sen_Σ y a cada morfismo de signatura la traducción $T(\sigma)$ entonces,

Lema: T es un functor.

Demostración: ...

El functor **Sen**



Sobre los modelos de diferentes signaturas

Acabamos de establecer una relación muy fuerte entre signaturas (como categoría) y sus conjuntos de sentencias (en la categoría Set)... ¿podremos hacer lo mismo con los modelos?

Sobre los modelos de diferentes signaturas

Acabamos de establecer una relación muy fuerte entre signaturas (como categoría) y sus conjuntos de sentencias (en la categoría Set)... ¿podremos hacer lo mismo con los modelos?

¿En qué categoría viven las categorías de modelos de las signaturas?

Sobre los modelos de diferentes signaturas

Acabamos de establecer una relación muy fuerte entre signaturas (como categoría) y sus conjuntos de sentencias (en la categoría Set)... ¿podremos hacer lo mismo con los modelos?

¿En qué categoría viven las categorías de modelos de las signaturas?

Por supuesto, ¡en la categoría de las categorías! **:-)**
(los objetos son categorías y los morfismos functores)

Sobre los modelos de diferentes signaturas

Acabamos de establecer una relación muy fuerte entre signaturas (como categoría) y sus conjuntos de sentencias (en la categoría Set)... ¿podremos hacer lo mismo con los modelos?

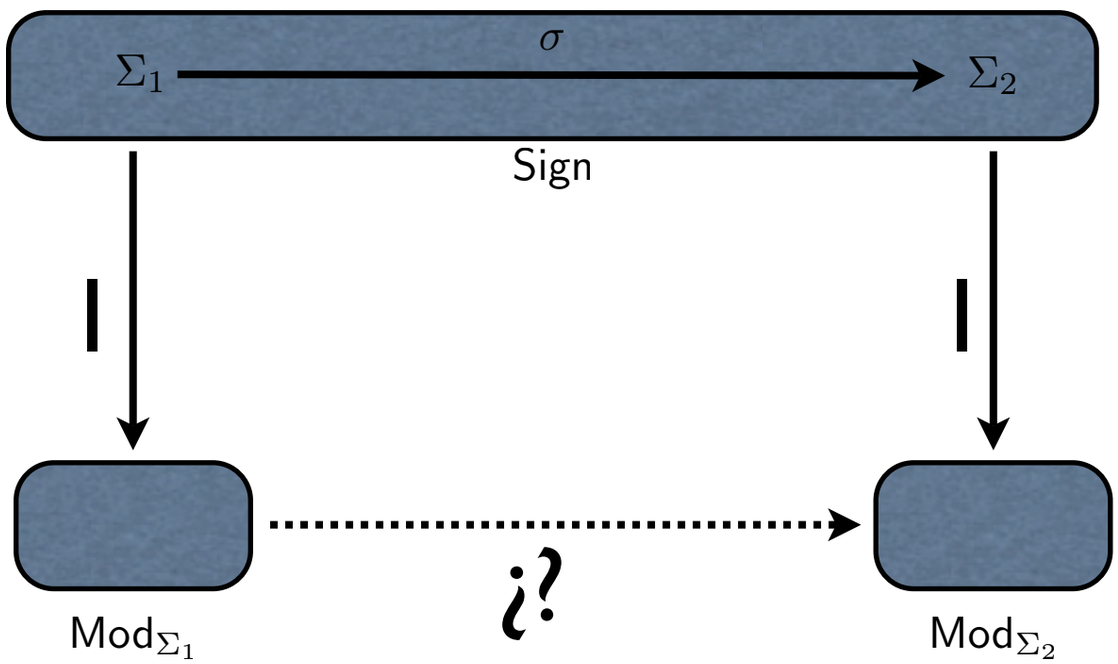
¿En qué categoría viven las categorías de modelos de las signaturas?

Por supuesto, ¡en la categoría de las categorías! **:-)**
(los objetos son categorías y los morfismos functores)

Sobre los modelos de diferentes signaturas

$$\Sigma_1 \xrightarrow{\quad\sigma\quad} \Sigma_2$$

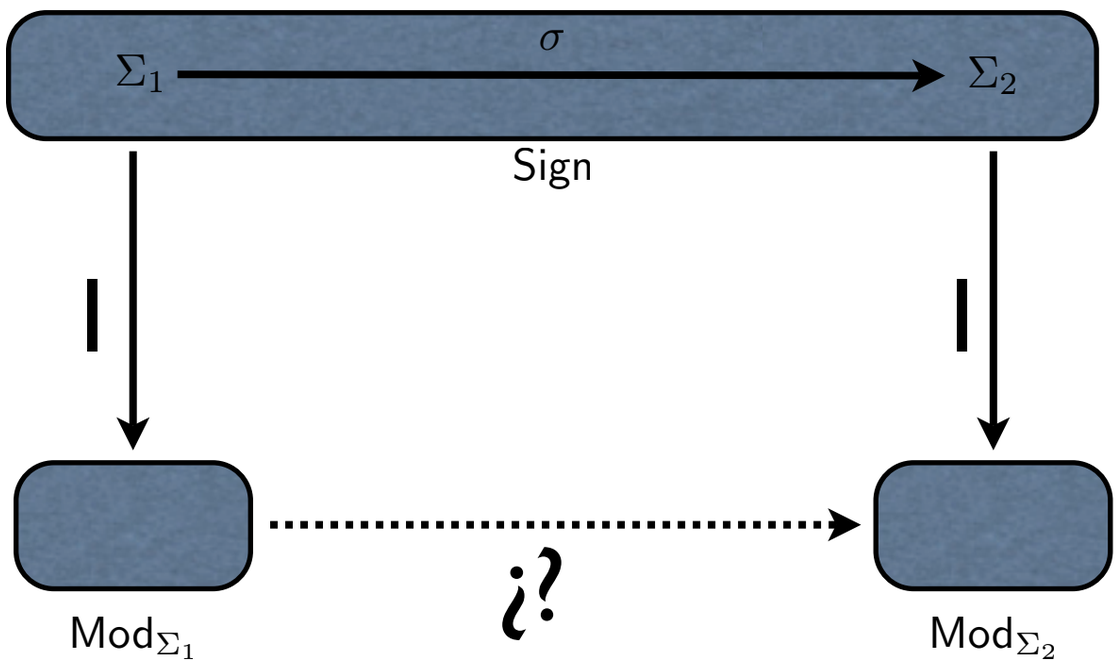
En Σ_2 podría haber más símbolos que en Σ_1. Esto implica un problema no menor; imaginen que queremos terminar el diagrama:



Sobre los modelos de diferentes signaturas

$$\Sigma_1 \xrightarrow{\quad\sigma\quad} \Sigma_2$$

En Σ_2 podría haber más símbolos que en Σ_1. Esto implica un problema no menor; imaginen que queremos terminar el diagrama:



¿Cómo haríamos para caracterizar a partir de modelos "chicos" sus correspondientes modelos "más grandes" que consigue interpretar signaturas con más símbolos?

Sobre los modelos de diferentes signaturas

¡Con magia! **Categoría opuesta** **[Fia05]**

Let $\mathsf{C} = \langle \mathcal{O}, \mathcal{A} \rangle$ be a category. Then C^{op} (the *opposite category* of C) is defined as $\langle \mathcal{O}, \mathcal{A}^{\mathsf{op}} \rangle$ where the morphisms in $\mathcal{A}^{\mathsf{op}}$ are the morphisms of $\mathcal{A}$ reverted (i.e. $f^{\mathsf{op}} : y \to x \in \mathcal{A}^{\mathsf{op}}$ if and only if $f : x \to y \in \mathcal{A}$) and such that for all $f, g \in G_1$ and $f \circ g \in G_2$, $(f \circ g)^{\mathsf{op}} = g^{\mathsf{op}} \circ f^{\mathsf{op}}$.

Sobre los modelos de diferentes signaturas

Fantástico... ¿y qué hago con una categoría opuesta?

La **solución** será establecer la relación entre las categorías de modelos, pero como "reflejo" de las relaciones en $\mathrm{Sign}^{\mathrm{op}}$.

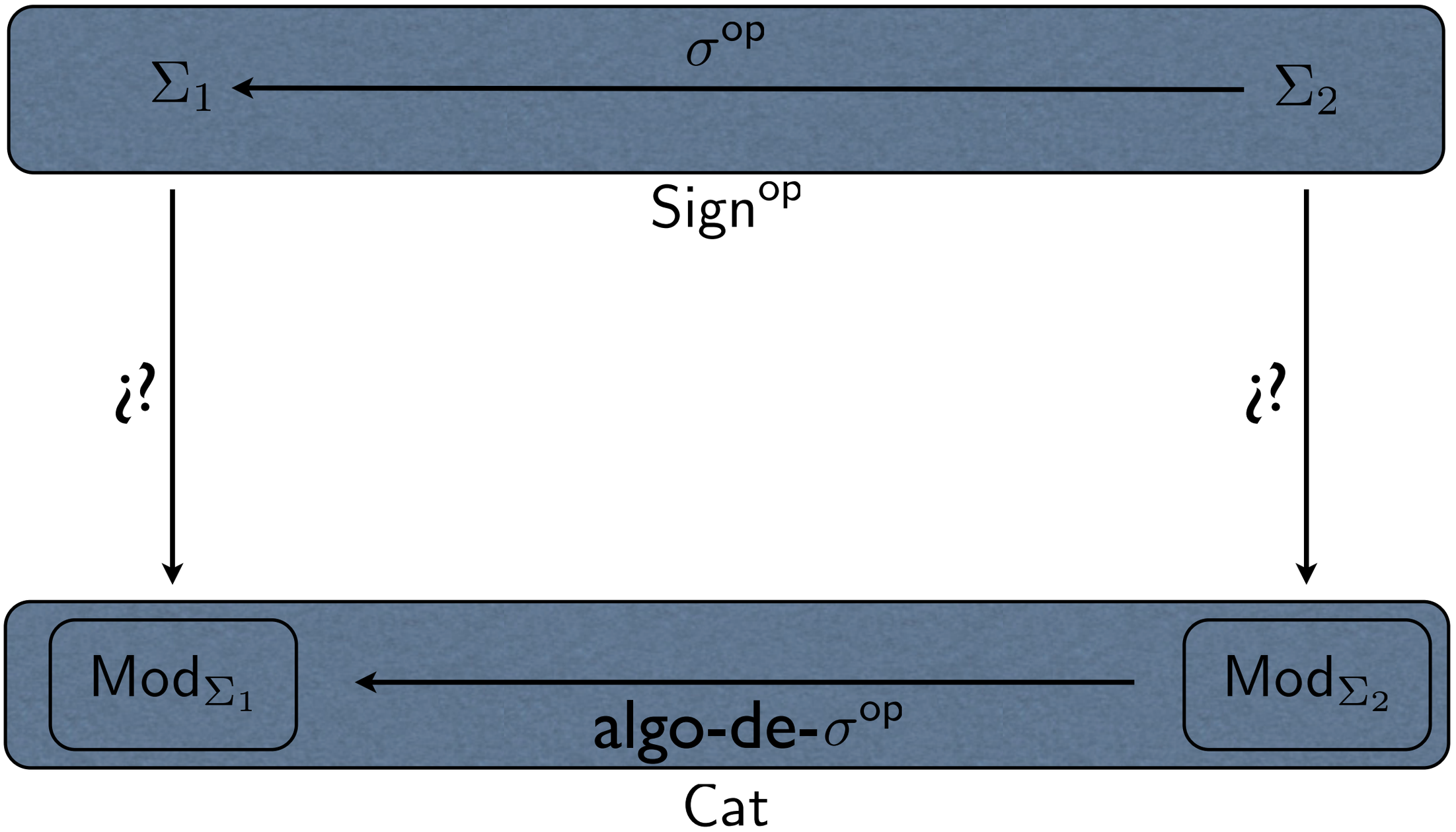
Sobre los modelos de diferentes signaturas

Fantástico... ¿y qué hago con una categoría opuesta?

El **problema** de los modelos era cómo caracteriza modelos "más grandes" a partir de modelos más "chicos", no?

La **solución** será establecer la relación entre las categorías de modelos, pero como "reflejo" de las relaciones en $\mathrm{Sign}^{\mathrm{op}}$.

Sobre los modelos de diferentes signaturas



Sobre los modelos de diferentes signaturas

Reducto de un álgebra [BS81]

Dados dos tipos T y T', tal que T está contenido en T', el T-reducto de un álgebra A de tipo T' es el álgebra B de tipo T tal que:

- los conjuntos coinciden, y
- para todo símbolo f de T, la interpretación de f en B es igual a la interpretación de f en A.

Sobre los modelos de diferentes signaturas

Un morfismo entre signaturas denota una relación entre tipos (de uno "más pobre" a uno "más rico"),

Luego, la relación entre los modelos estará determinada por tomar reducto de la clase de algebras correspondiente a la signatura "más rica" de acuerdo al tipo de la signatura "más pobre".

Sobre los modelos de diferentes signaturas

Ejemplo

TAD pNat
genero: pnat
usa: bool
exporta: pnat, p0, psuc, p+
generadores:
p0: $\longrightarrow$ pnat
psuc: pnat $\longrightarrow$ pnat
observadores:
pes0?: pnat $\longrightarrow$ bool
ppred: x: pnat $\longrightarrow$ pnat
otras operaciones:
p+: nat x nat $\longrightarrow$ nat

Fin TAD

σ^{op}

TAD Nat
genero: nat
usa: bool
exporta: nat, 0, suc, +, *
generadores:
0: $\longrightarrow$ nat
suc: nat $\longrightarrow$ nat
observadores:
es0?: nat $\longrightarrow$ bool
pred: x: nat $\longrightarrow$ nat
otras operaciones:
+: nat x nat $\longrightarrow$ nat
*: nat x nat $\longrightarrow$ nat

Fin TAD

Sobre los modelos de diferentes signaturas

Ejemplo

Luego, la estructura:

$$\langle \mathbb{N}, 0, +1, -1, + \rangle$$

que resulta ser el pNat-reducto del modelo de Nat:

$$\langle \mathbb{N}, 0, +1, -1, +, \cdot \rangle$$

Es modelo de pNat.

Sobre los modelos de diferentes signaturas

Reducto de un álgebra

[BS81]

Let $\Sigma, \Sigma' \in |\mathsf{Sign}|$ and $\sigma : \Sigma \to \Sigma'$ be a morphism in Sign. Let $\mathcal{M}' \in Mod_{\Sigma'}$, then

$$\mathcal{M}' \restriction_{\sigma^{\mathsf{op}}} = \langle \mathcal{P}', \{ f^{\mathcal{M}'}_{\sigma(i)} \}_{i \in \mathcal{I}} \rangle \ .$$

¿Quién es algo-de-σ^{op}?

algo-de-σ^{op} relaciona las categorías de modelos correspondientes a las signaturas involucradas en σ^{op} y por lo tanto es...

¿Quién es algo-de-σ^{op}?

algo-de-σ^{op} relaciona las categorías de modelos correspondientes a las signaturas involucradas en σ^{op} y por lo tanto es... **¡un functor!**

¿Quién es algo-de-σ^{op}?

algo-de-σ^{op} relaciona las categorías de modelos correspondientes a las signaturas involucradas en σ^{op} y por lo tanto es... **¡un functor!**

El functor $\mathbf{Mod}(\sigma^{\mathsf{op}})$

Let $\Sigma, \Sigma' \in |\mathsf{Sign}|$ and $\sigma : \Sigma \to \Sigma'$ be a morphism in Sign. Let $\mathcal{M}' \in |\mathbf{Mod}(\Sigma')|$ and γ' be a morphism in $\mathbf{Mod}(\Sigma')$, then

- $\mathbf{Mod}(\sigma^{\mathsf{op}})(\mathcal{M}') = \mathcal{M}' \restriction_{\sigma^{\mathsf{op}}}$,
- $\mathbf{Mod}(\sigma^{\mathsf{op}})(\gamma') = \gamma'$.

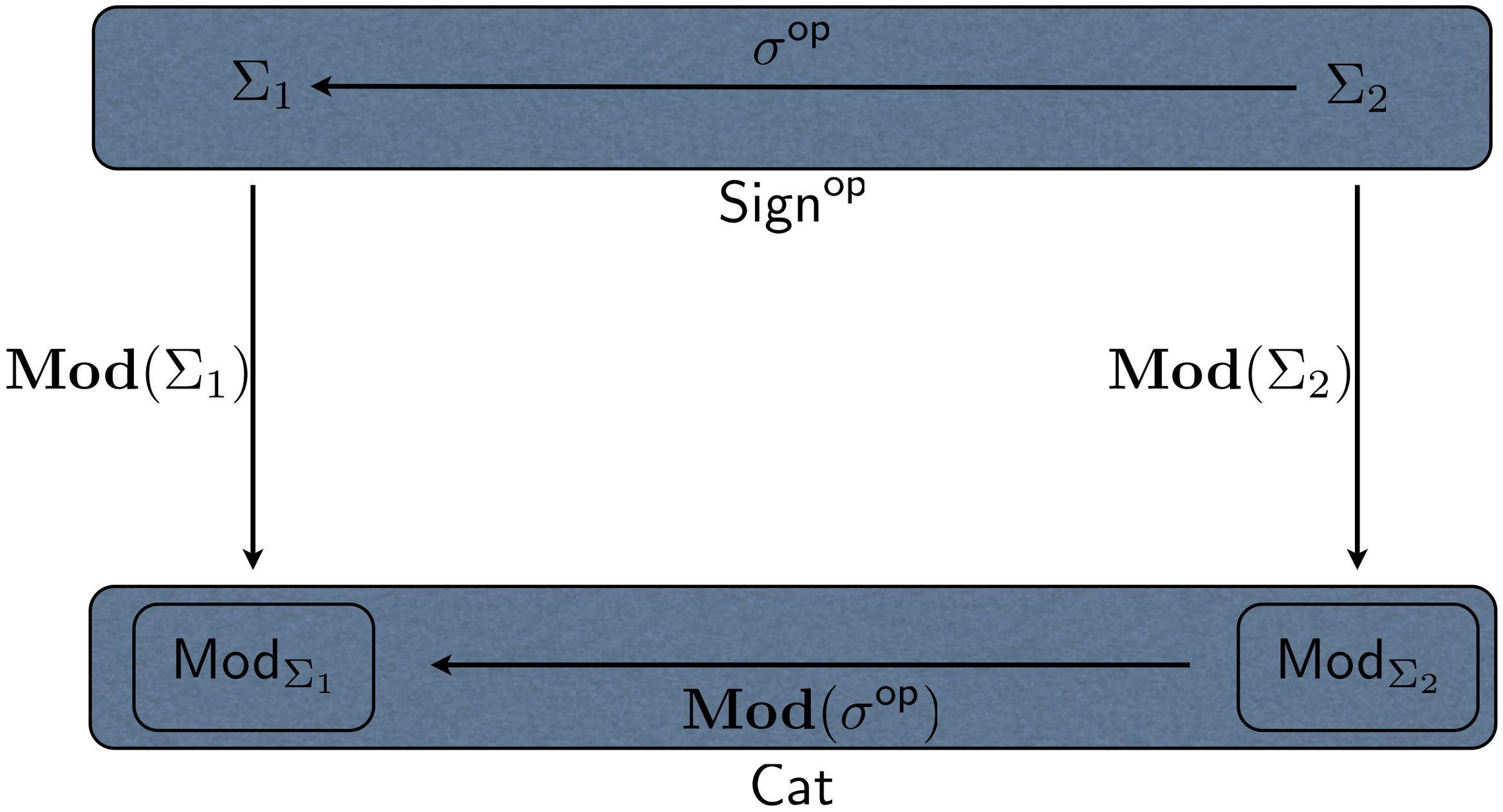
El functor **Mod**

Podemos definir $\mathbf{Mod}(\Sigma) = Mod_\Sigma$ y ya sabemos que $\mathbf{Mod}(\sigma^{\mathrm{op}})$ es un functor entre las categorías de modelos correspondientes a las signaturas involucradas en σ^{op}, luego, sólo falta...

Lema: $\mathrm{Mod} : \mathrm{Sign}^{\mathrm{op}} \to \mathrm{Cat}$ es un functor.

Demostración: ...

El functor **Mod**



Final del viaje

Instituciones [GB84, GB92]

An *institution* is a structure of the form $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ satisfying the following conditions:

- Sign is a category of signatures,
- $\mathbf{Sen} : \mathsf{Sign} \to \mathsf{Set}$ is a functor (let $\Sigma \in |\mathsf{Sign}|$, then $\mathbf{Sen}(\Sigma)$ returns the set of Σ-sentences),
- $\mathbf{Mod} : \mathsf{Sign}^{\mathsf{op}} \to \mathsf{Cat}$ is a functor (let $\Sigma \in |\mathsf{Sign}|$, then $\mathbf{Mod}(\Sigma)$ returns the category of Σ-models),
- $\{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}$, where $\models^{\Sigma} \subseteq |\mathbf{Mod}(\Sigma)| \times \mathbf{Sen}(\Sigma)$, is a family of binary relations,

and for any signature morphism $\sigma : \Sigma \to \Sigma'$, Σ-sentence $\phi \in \mathbf{Sen}(\Sigma)$ and Σ'-model $\mathcal{M}' \in |\mathbf{Mod}(\Sigma)|$ the following $\models$-invariance condition holds:

$$\mathcal{M}' \models^{\Sigma'} \mathbf{Sen}(\sigma)(\phi) \;\text{ iff }\; \mathbf{Mod}(\sigma^{\mathsf{op}})(\mathcal{M}') \models^{\Sigma} \phi \ .$$

Composicionalidad y propiedades emergentes

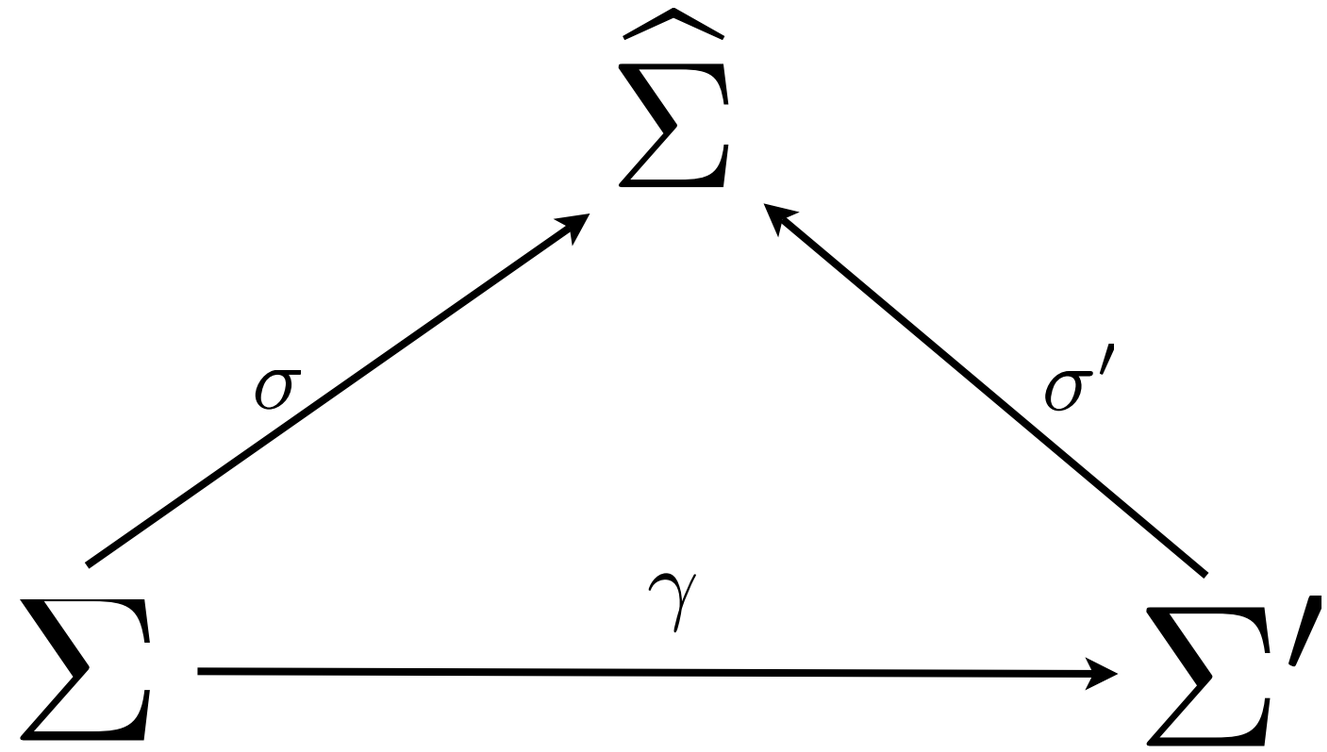
Carlos G. Lopez Pombo
Departamento de computación, FCEyN, UBA
y CONICET

Hasta ahora...

... lo único que hemos conseguido es caracterizar, en teoría de categorías qué es una lógica (porque me creyeron que iba a ser útil)

¿En qué nos ayudará esto para enfrentar el problema de la composición de especificaciones?

¿Qué nos gustaría?



Saber:

- quién es $\widehat{\Sigma}$, y
- quienes son σ y σ'.

tal que $\widehat{\Sigma}$ expresa la relación inducida por γ.

Más teoría de categorías

Diagrama [Fia05]

Let C be a category and I a graph. A *diagram* with shape I in C is a graph homomorphism $\delta = \langle \delta_0, \delta_1 \rangle : I \to graph(\mathsf{C})$.

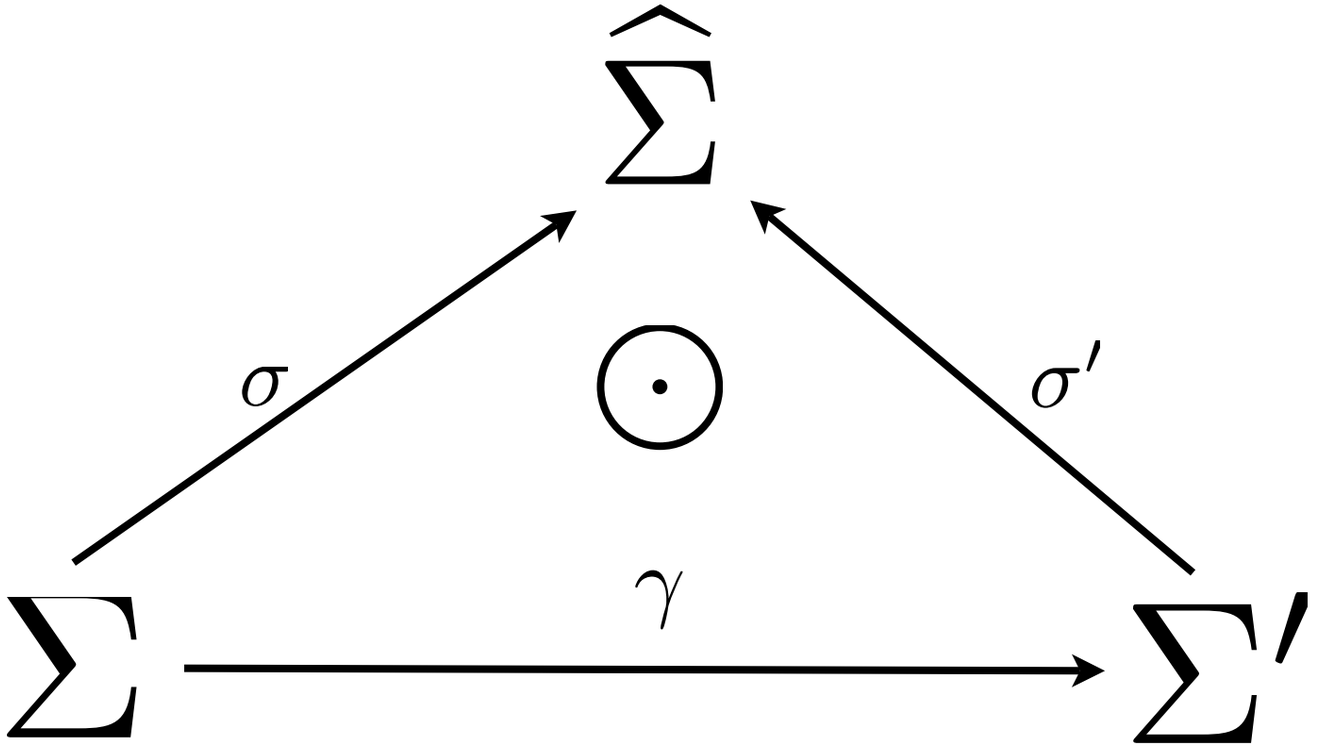
Diagrama conmutativo [Fia05]

Let C be a category and $I = \langle G_0, G_1 \rangle$ a graph. Then, a diagram $\delta = \langle \delta_0, \delta_1 \rangle : I \to graph(\mathsf{C})$ commutes if and only if for $f_1, \ldots, f_i \in G_i$, $g_1, \ldots, g_j \in G_j$ such that $src(f_1) = src(g_1)$ and $trg(f_i) = trg(g_j)$, then

$$\delta_1(f_1) \circ \cdots \circ \delta_1(f_i) = \delta_1(g_1) \circ \cdots \circ \delta_1(g_j)$$

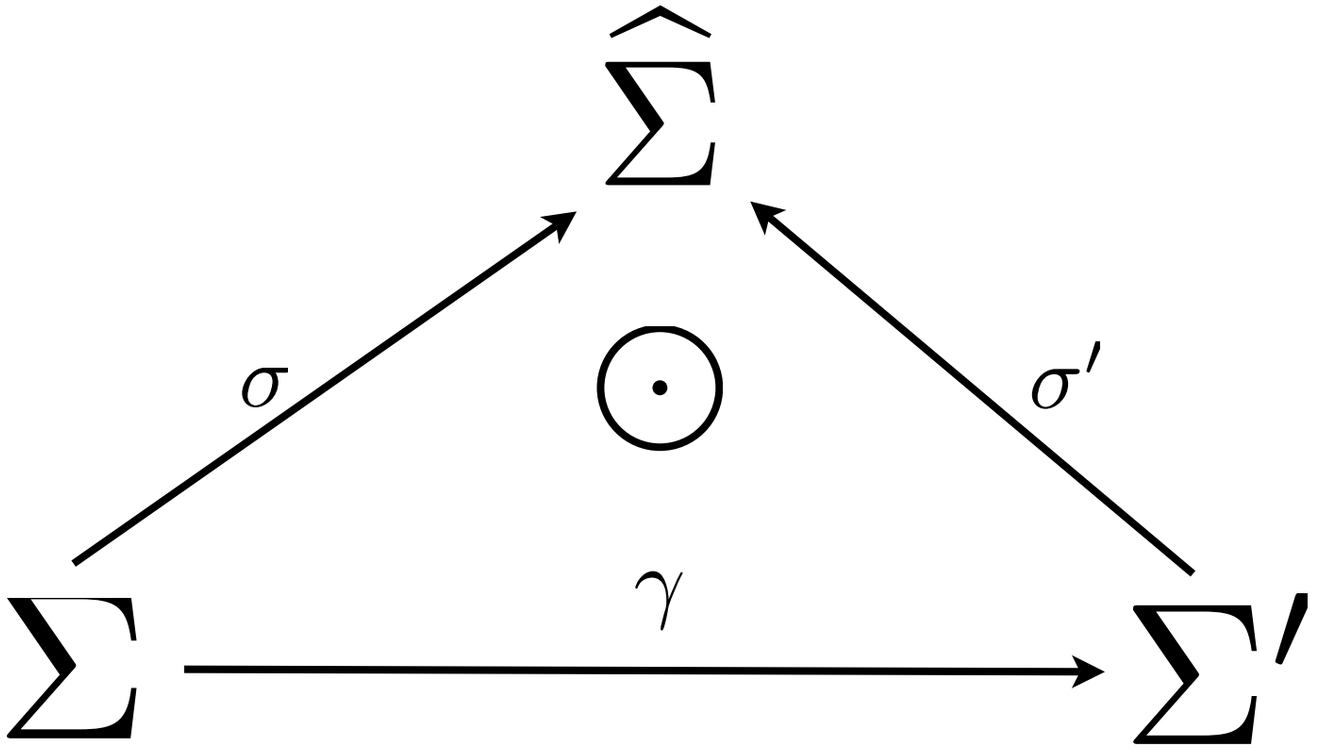
holds in C.

Más teoría de categorías



¿Qué significa que este diagrama conmute?

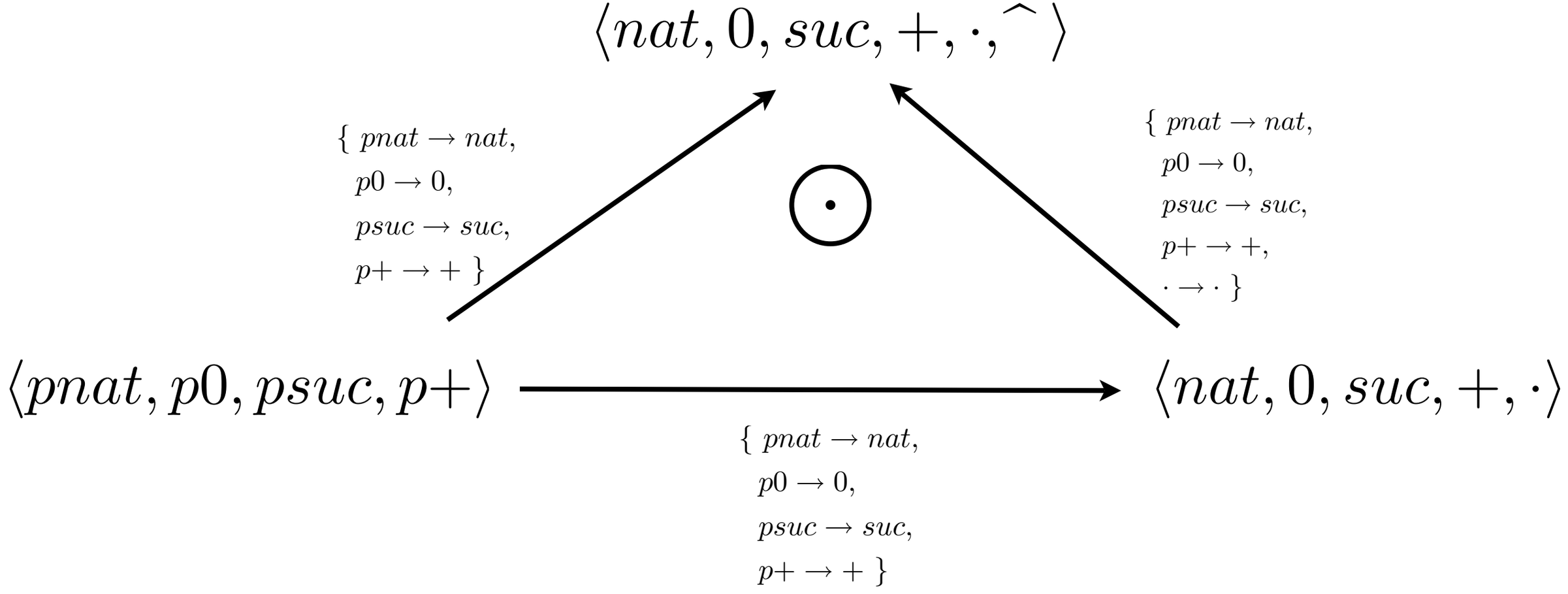
Más teoría de categorías



¿Qué significa que este diagrama conmute?

¡Justo lo que necesitamos!... que traducir símbolos directamente de Σ a $\widehat{\Sigma}$ es lo mismo que primero traducir a Σ' y luego a $\widehat{\Sigma}$.

Más teoría de categorías



Más aun...

Conos y co-conos

[Fia05]

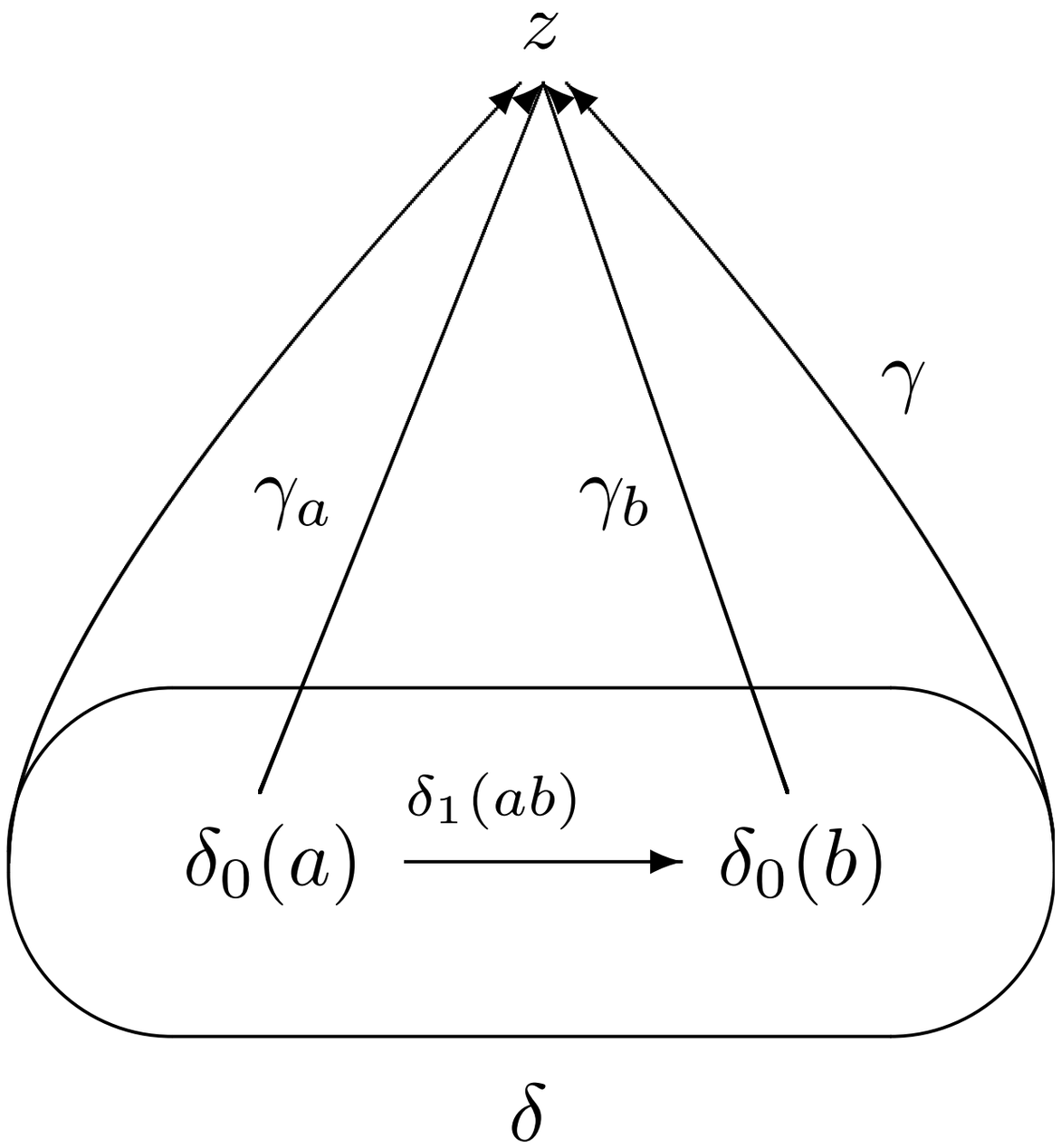
Let C be a category, $I = \langle G_0, G_1 \rangle$ be a graph, and $\delta = \langle \delta_0, \delta_1 \rangle$ be a diagram $\delta : I \to graph(\mathsf{C})$. A *co-cone* with base δ is an object $z \in |\mathsf{C}|$ (the vertex of the co-cone), and a family of morphisms $\{\gamma_{\delta_0(a)} : \delta_0(a) \to z\}_{a \in \delta_0(G_0)}$ (the edges of the co-cone), usually denoted $\gamma : \delta \to z$.

A co-cone is *commutative* if and only if for any pair of vertexes $a, b \in G_0$ such that $\langle a, b \rangle \in G_1$, $\delta_1(\langle a, b \rangle) \circ \gamma_b = \gamma_a$.

The concept of *cone* is the dual notion of co-cone (i.e. the equivalent definition but reversing the arrows).

Más aun...

Conos y co-conos



Secuencias

TAD Seq
genero: seq
usa: bool, nat
exporta: ...
generadores:

[]:		seq
&&: seq x T		seq

observadores:

vacía?:	seq	bool	
prim:	x: seq	T	not vacía? (x)
resto:	x: seq	seq	not vacía? (x)

otras operaciones:

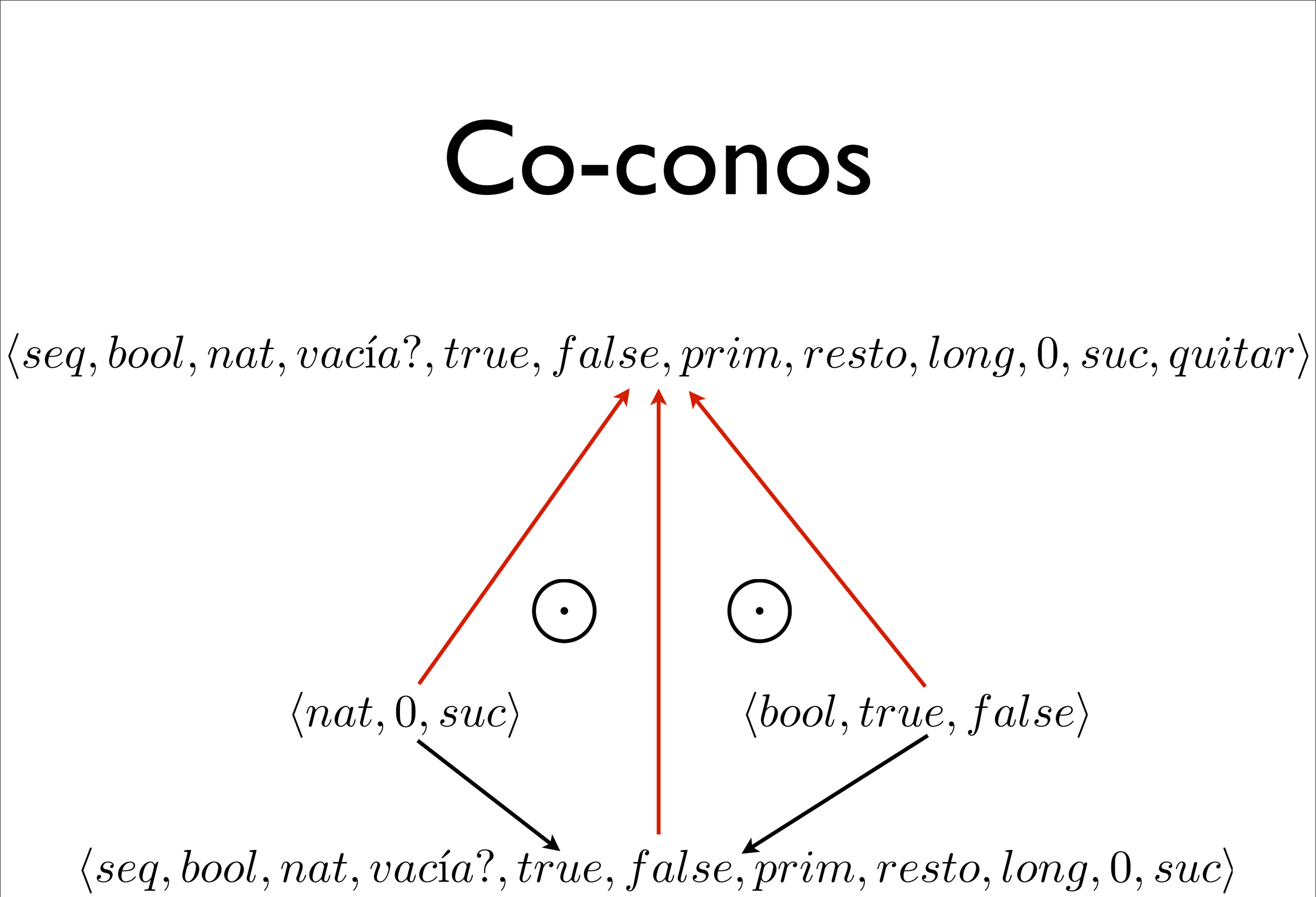
long:	seq	nat

axiomas:

...

Fin TAD

Co-conos


⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc, quitar⟩
⟨nat, 0, suc⟩
⟨bool, true, false⟩
⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc⟩

Más aun...

Límites y co-límites

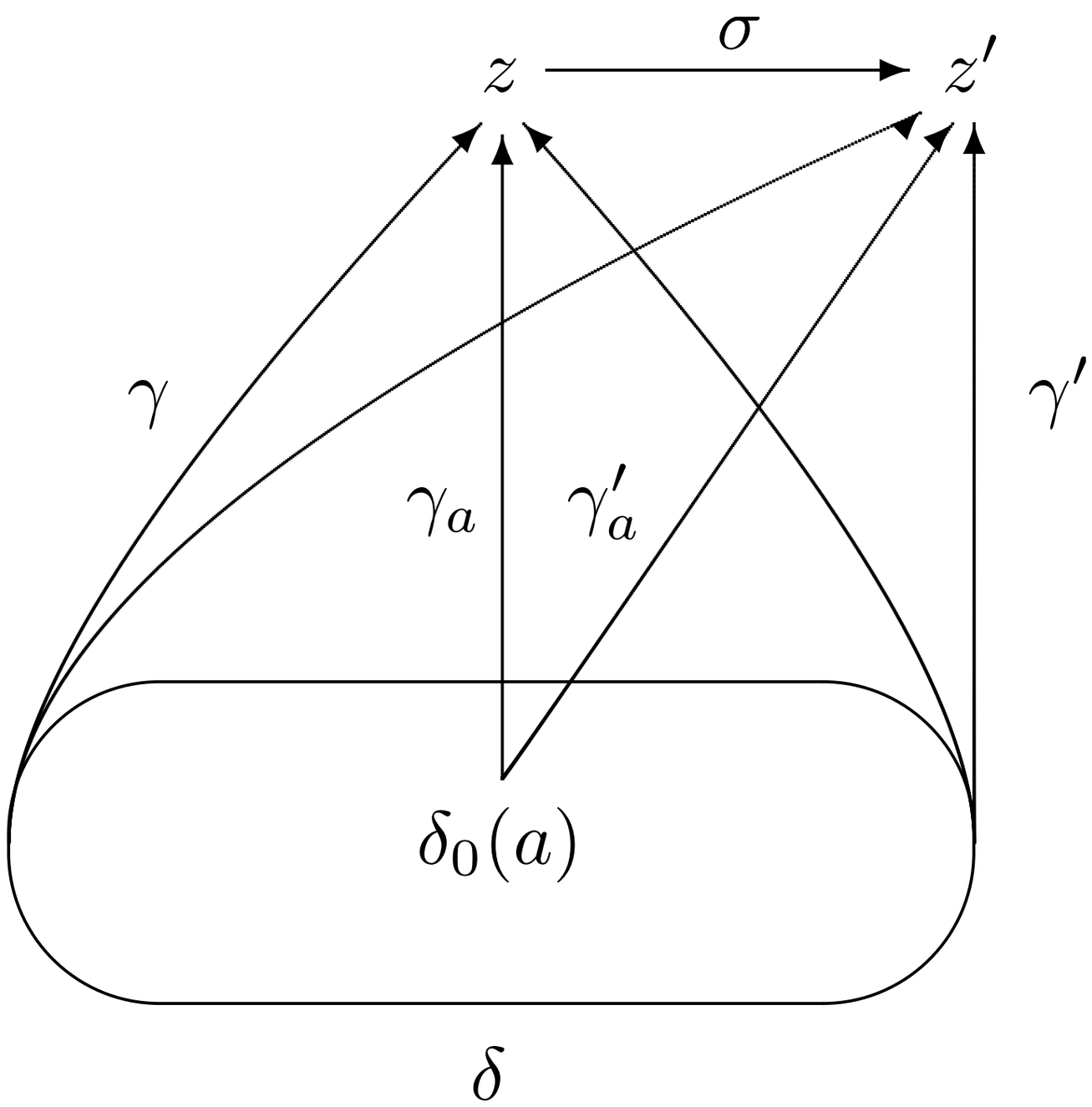
[Fia05]

Let C be a category, $I = \langle G_0, G_1 \rangle$ be a graph, and $\delta = \langle \delta_0, \delta_1 \rangle$ be a diagram $\delta : I \to graph(\mathsf{C})$. A *co-limit* is a commutative co-cone $\gamma : \delta \to z$ such that for every commutative co-cone $\gamma' : \delta \to z'$, there is a unique morphism $\sigma : z \to z'$ satisfying $\gamma \circ \sigma = \gamma'$ (i.e. for all $a \in G_0$, $\gamma_a \circ \sigma = \gamma'_a$).

The concept of *limit* is the dual notion of co-limit (i.e. the equivalent definition but reversing the arrows).

Más aun...

Límites y co-límites



Más aun...

Completitud y co-completitud [Fia05]

A category is (finitely) co-complete if and only if all (finite) diagrams have co-limits.

A category is (finitely) complete if and only if all (finite) diagrams have limits.

Más aun...

Objetos iniciales y terminales [Fia05]

Let C be a category and $x \in |\mathsf{C}|$. Then, x is *initial* if and only if for all $y \in |\mathsf{C}|$, there exists a unique morphism $f : x \to y$.

Analogously, x is *terminal* if and only if for all $y \in |\mathsf{C}|$, there exists a unique morphism $f : y \to x$.

Nota: $\emptyset$ es el objeto inicial de Set.

Más aun...

Pushouts y pullbacks

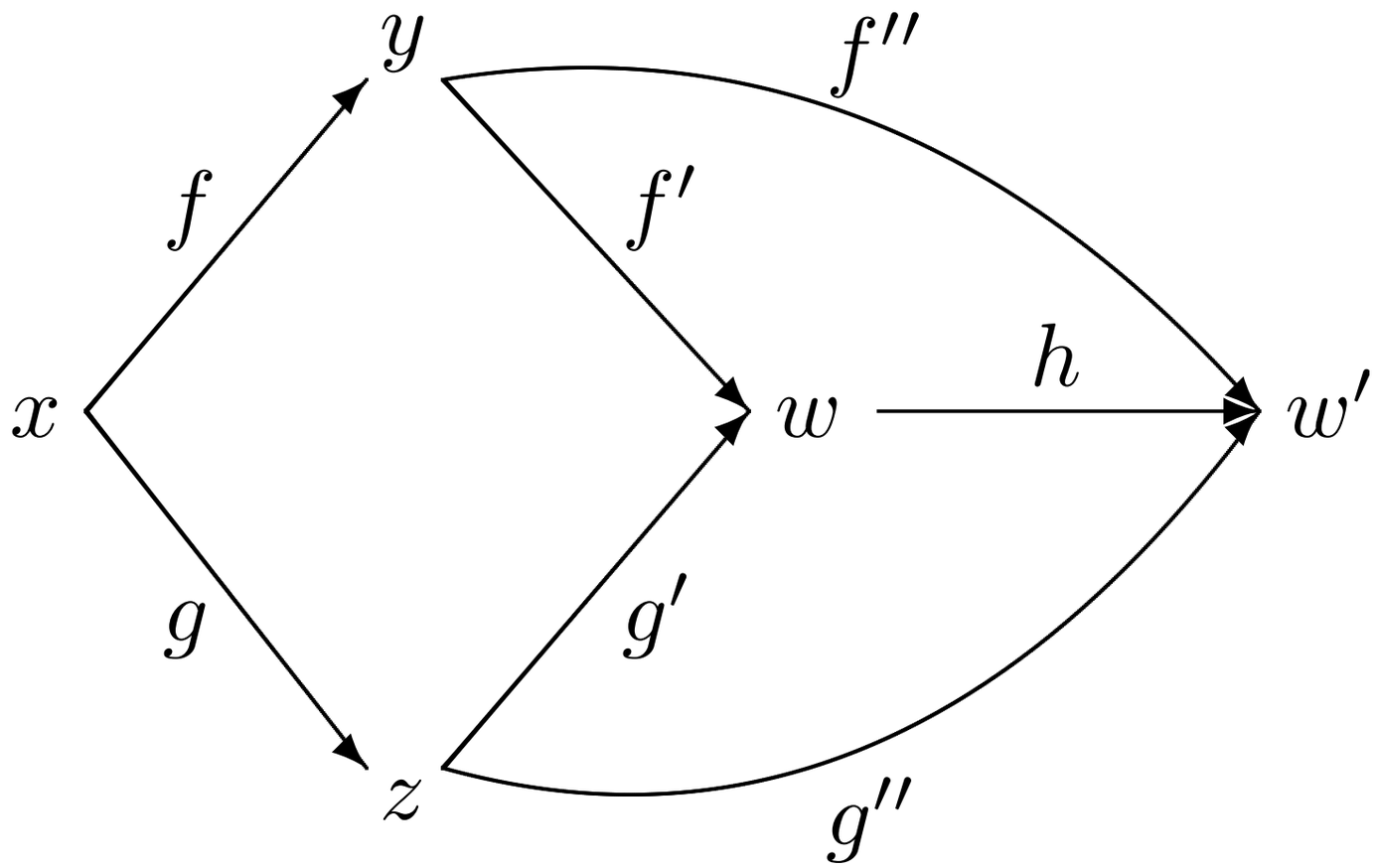
[Fia05]

Let C be a category and $f : x \to y$, $g : x \to z$ morphisms in C, then a *pushout* of f and g consists of $f' : y \to w$, $g' : z \to w$ morphisms in C such that:

- $f \circ f' = g \circ g'$, and
- for any other $f'' : y \to w'$, $g'' : z \to w'$ morphisms in C such that $f \circ f'' = g \circ g''$, there exists a unique $h : w \to w'$ morphism in C such that $f' \circ h = f''$ and $g' \circ h = g''$.

Pullback are defined analogously to pushouts but considering the arrows in the opposite direction.

Pushouts



Co-completitud y completitud

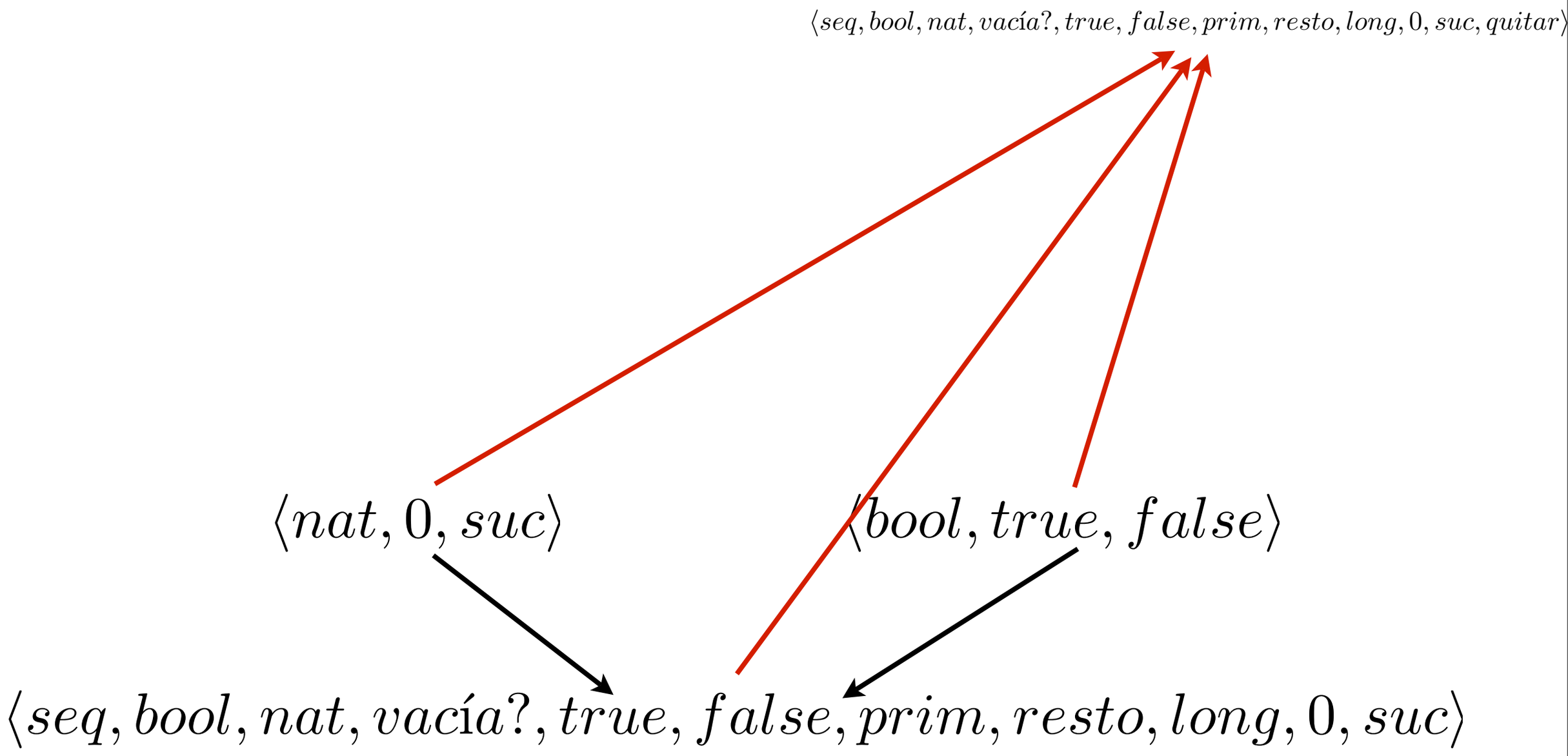
A category is finitely co-complete if and only if it has initial objects and pushouts of all pairs of morphisms with common source.

A category is finitely complete if and only if it has final objects and pullbacks of all pairs of morphisms with common source.

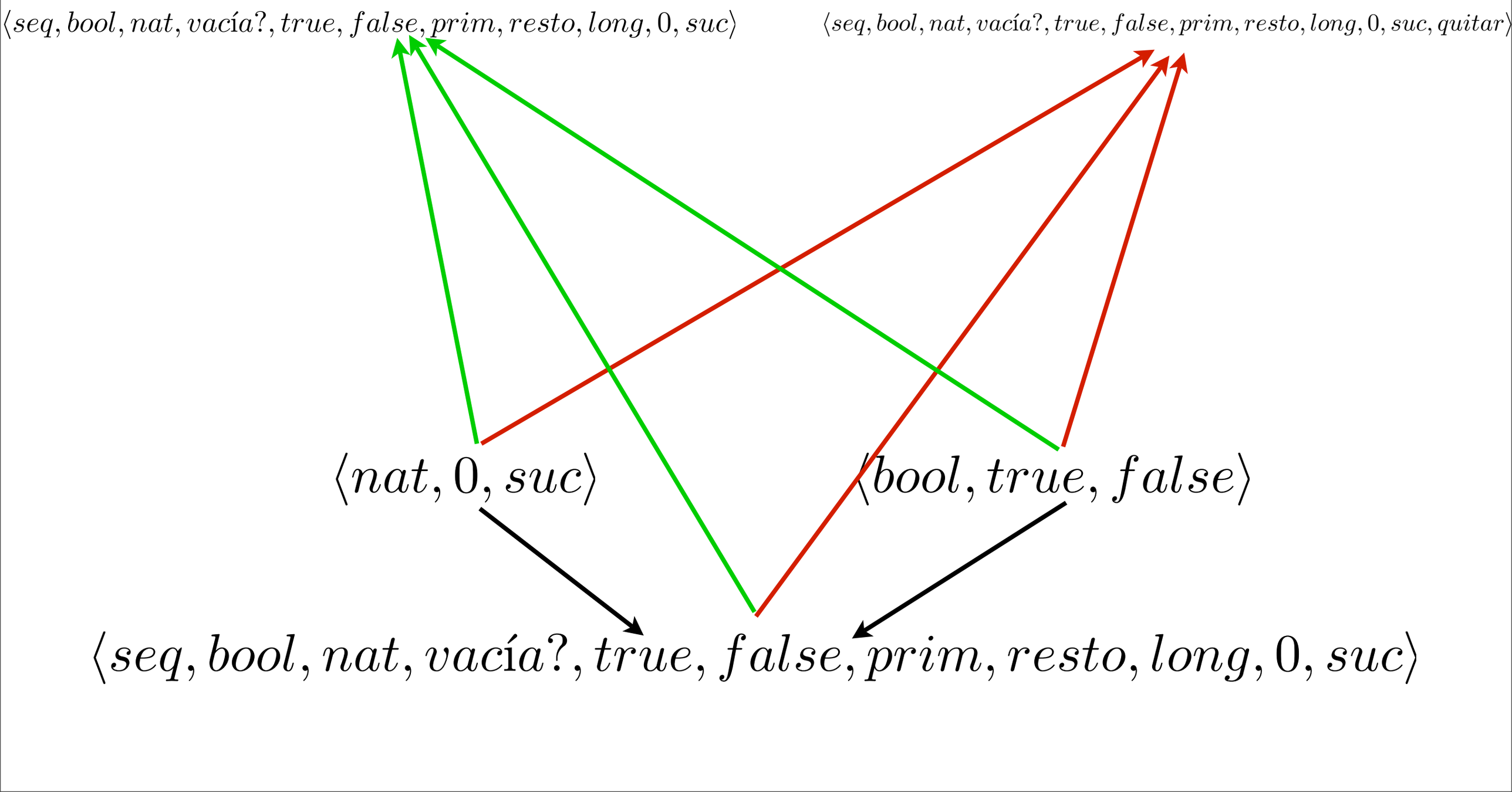
Lema: Sign es finitely co-complete.

Demostración: ...

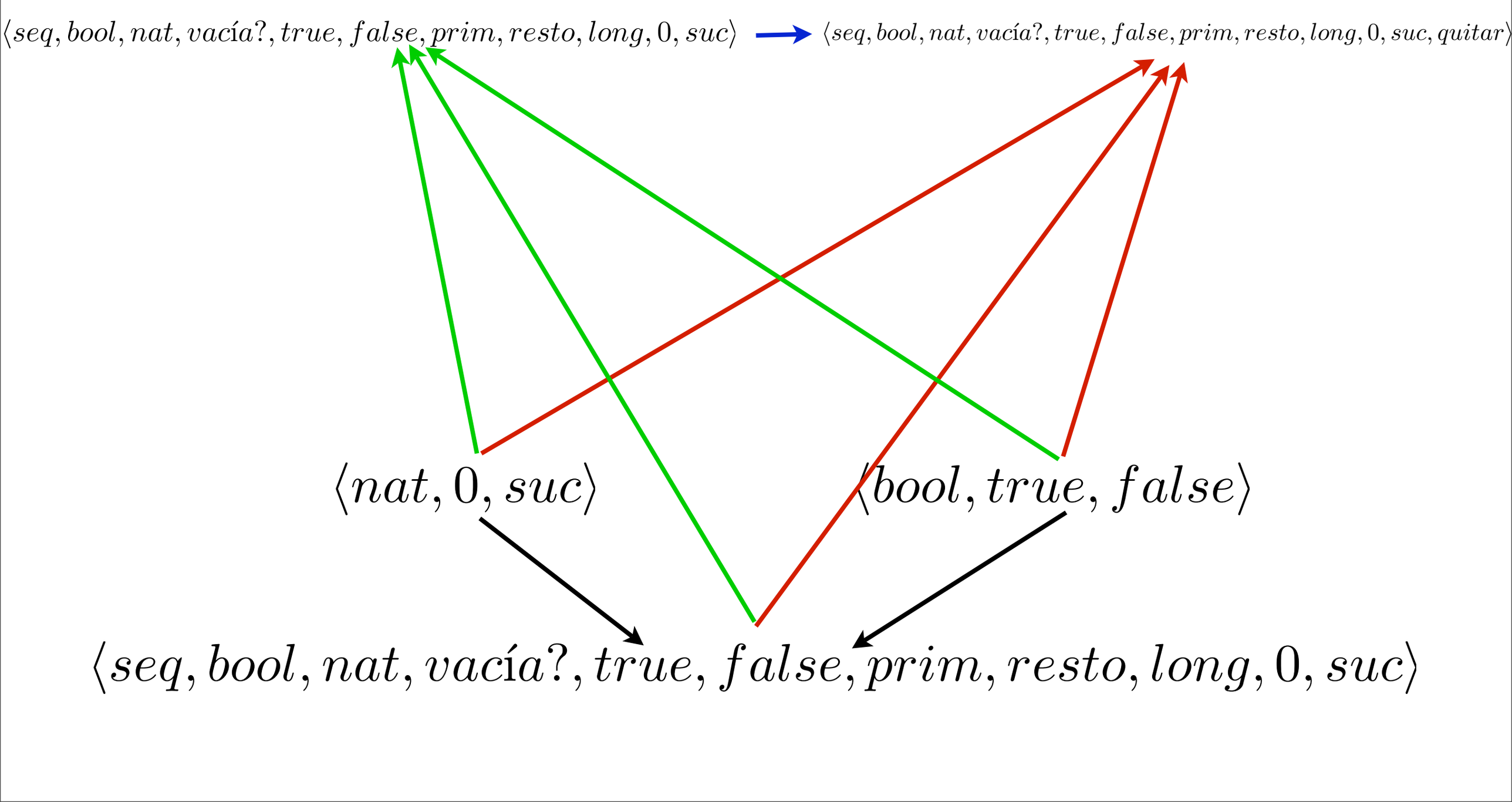
Co-límites


⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc, quitar⟩
⟨nat, 0, suc⟩
⟨bool, true, false⟩
⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc⟩

Co-límites


⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc⟩
⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc, quitar⟩
⟨nat, 0, suc⟩
⟨bool, true, false⟩
⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc⟩

Co-límites


⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc⟩
⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc, quitar⟩
⟨nat, 0, suc⟩
⟨bool, true, false⟩
⟨seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc⟩

Mentiras piadosas

Si miramos el diagrama anterior observaremos que para poder construir el co-límite hemos utilizado una signatura para **nat** y para **bool** que no son las reales puesto que hay muchos símbolos que no están presentes.

$$\langle nat, 0, suc \rangle \qquad \langle bool, true, false \rangle$$

cuando es realidad querríamos usar:

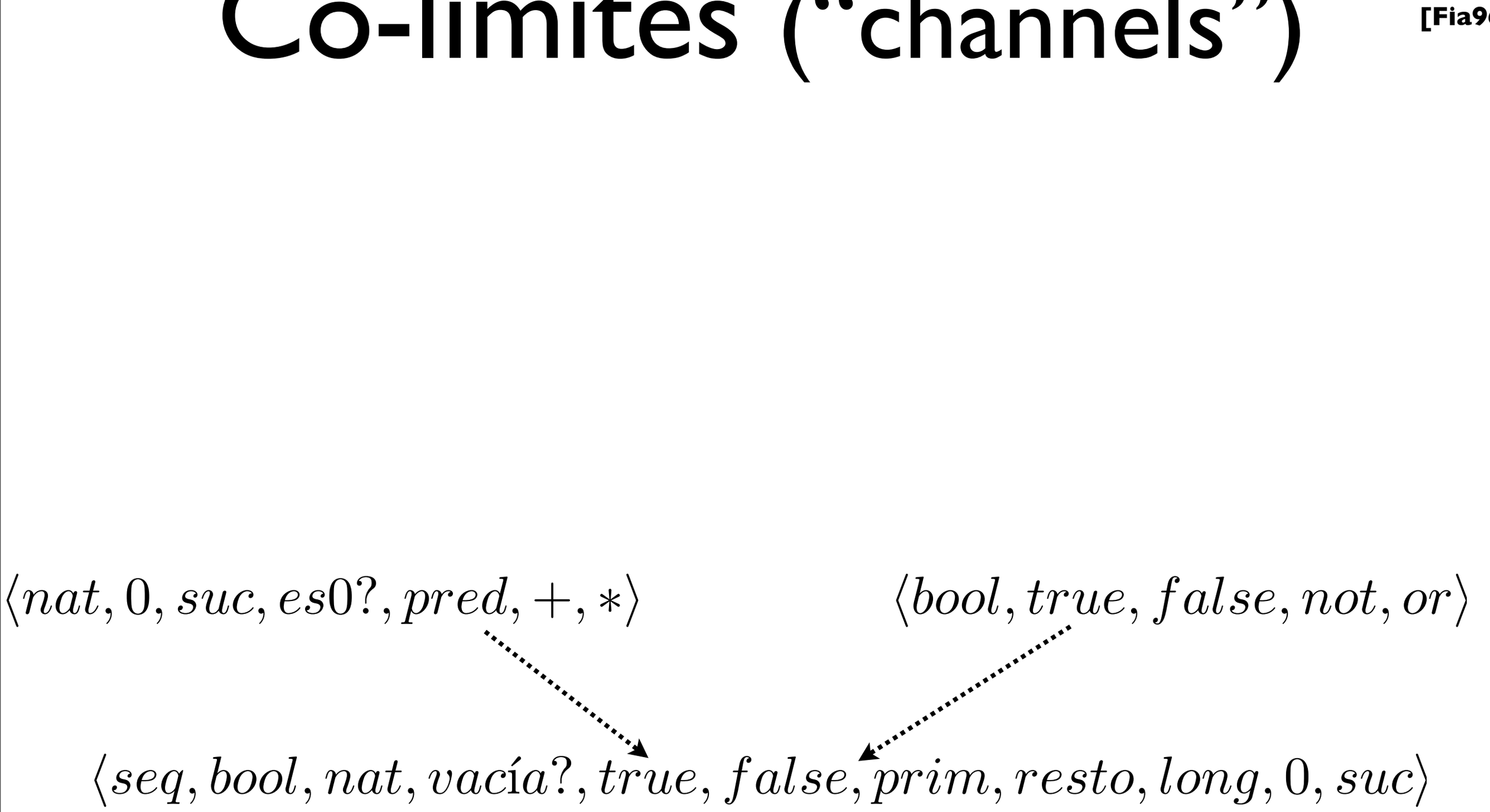
$$\langle nat, 0, suc, es0?, pred, +, * \rangle \qquad \langle bool, true, false, not, or \rangle$$

Co-límites ("channels") [Fia96]

$\langle nat, 0, suc, es0?, pred, +, * \rangle$ $\langle bool, true, false, not, or \rangle$

$\langle seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc \rangle$

Co-límites ("channels") [Fia96]

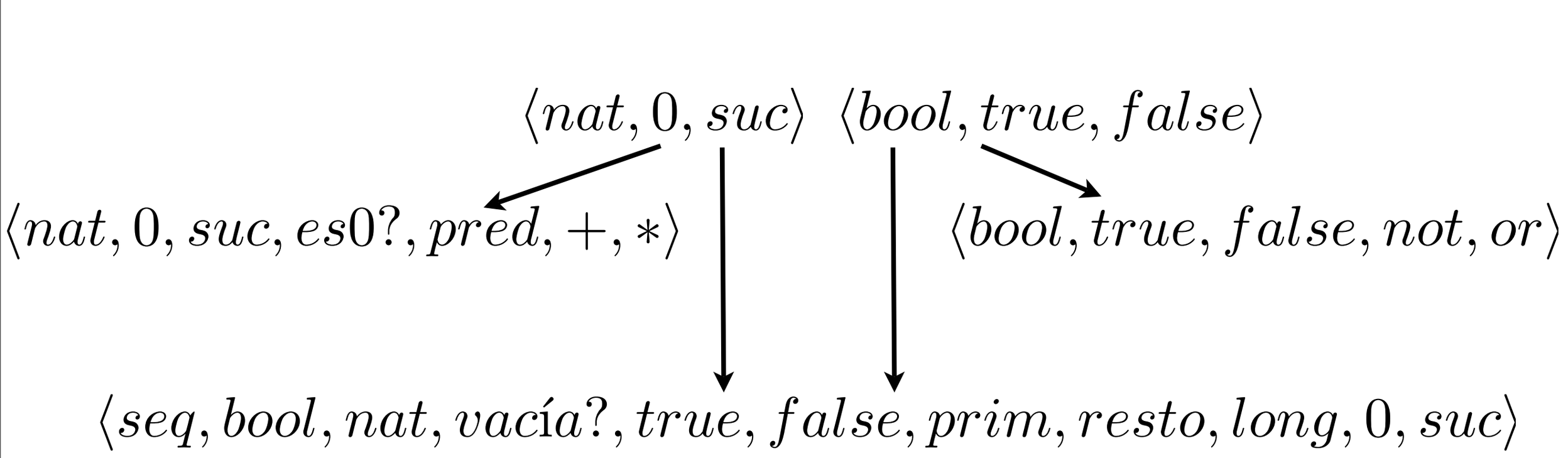


Co-límites ("channels") [Fia96]

$\langle nat, 0, suc, es0?, pred, +, * \rangle$ $\langle bool, true, false, not, or \rangle$

$\langle seq, bool, nat, vacía?, true, false, prim, resto, long, 0, suc \rangle$

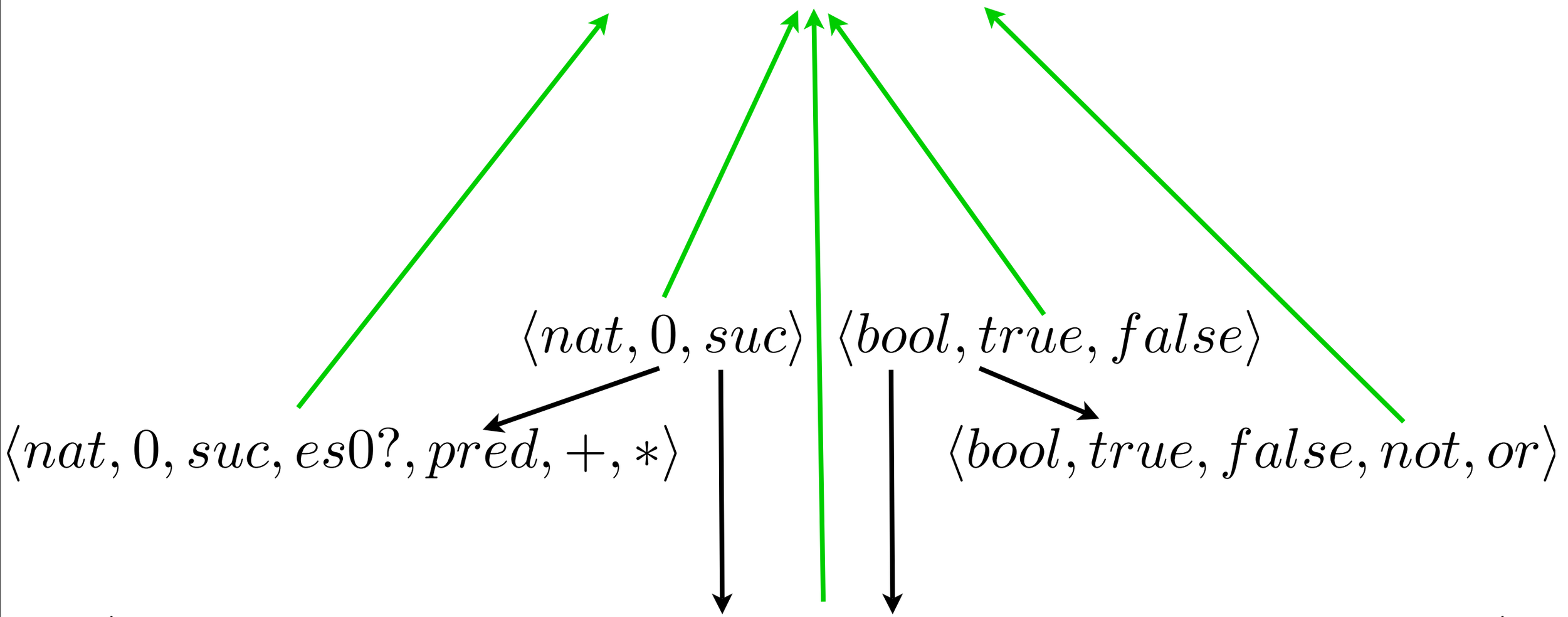
Co-límites ("channels") [Fia96]



Nota: las signaturas $\langle nat, 0, suc\rangle$ y $\langle bool, true, false\rangle$ se denominan "channels".

Co-límites ("channels") [Fia96]

$\langle seq, bool, nat, vacía?, true, false, not, or, prim, resto, long, 0, suc, es0?, pred, +, * \rangle$



Nota: las signaturas $\langle nat, 0, suc \rangle$ y $\langle bool, true, false \rangle$ se denominan "channels".

Con gusto a poco

Nuestro actual concepto de lógica hace foco en las (relaciones entre) signaturas, pero una especificación es más que eso, es una presentación de una teoría puesto que incorpora axiomas para las operaciones.

La categoría de las teorías

[Mes89]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ be an institution, then Th, its category of theories, is a pair $\langle \mathcal{O}, \mathcal{A} \rangle$ such that:

- $\mathcal{O} = \{\langle \Sigma, \Gamma \rangle \mid \Sigma \in |\mathsf{Sign}| \text{ and } \Gamma \subseteq \mathbf{Sen}(\Sigma)\}$, and
- $\mathcal{A} = \left\{ \langle \sigma : (\Sigma, \Gamma) \to (\Sigma', \Gamma') \mid \begin{array}{l} \langle \Sigma, \Gamma \rangle, \langle \Sigma', \Gamma' \rangle \in \mathcal{O}, \\ \sigma : \Sigma \to \Sigma' \text{ is a morphism in } \mathsf{Sign} \text{ and} \\ \text{for all } \gamma \in \Gamma, \Gamma' \models^{\Sigma'} \mathbf{Sen}(\sigma)(\gamma) \end{array} \right\}$.

In addition, if a morphism $\sigma : (\Sigma, \Gamma) \to (\Sigma', \Gamma')$ satisfies $\mathbf{Sen}(\sigma)(\Gamma) \subseteq \Gamma'$ it is called *axiom preserving*. This defines the category Th_0 by keeping only those morphisms of Th that are axiom preserving. It is easy to notice that $\mathsf{Th}_0 \hookrightarrow \mathsf{Th}$. Now, if we consider the definition of $\mathbf{Mod}$, extended to signatures and set of sentences, we get a functor $\mathbf{Mod} : \mathsf{Th}^{\mathsf{op}} \to \mathsf{Cat}$ defined as follows: let $T = \langle \Sigma, \Gamma \rangle \in |\mathsf{Th}|$, then

$$\mathbf{Mod}(T) = \mathbf{Mod}_{\langle \Sigma, \Gamma \rangle} \ .$$

La categoría de las teorías

[Mes89]

Lema: Th y Th0 son finitely co-complete.

Demostración: ...

En la categoría de las signaturas

Let $G = \langle V, E \rangle$ a graph and $\delta : G \to \mathsf{Sign}$ a finite diagram in Sign. Let $\Sigma_v = \langle \{f_i\}_{i \in \mathcal{I}_v} \rangle \in |\mathsf{Sign}|$ for all $v \in V$. Then, the co-cone $\langle \langle S \rangle, \{\gamma_v : \Sigma_v \to \langle S \rangle\}_{v \in V} \rangle$ where:

- $S = \left(\bigcup_{v \in V} \{f_i\}_{i \in \mathcal{I}_v}\right) \mid \overset{*}{\longleftrightarrow}^{\delta}$, and
- $\gamma_v(f_i) = [\![f_i]\!]_{\overset{*}{\longleftrightarrow}^{\delta}}$ for all $i \in \mathcal{I}_v$,

is a co-limit of δ.

$\overset{*}{\longleftrightarrow}^{\delta}$ denota la clausura reflexiva y transitiva de δ.

$[\![\mathrm{f}_i]\!]_{\overset{*}{\longleftrightarrow}^{\delta}}$ denota la clase de equivalencia de f_i bajo $\overset{*}{\longleftrightarrow}^{\delta}$.

En la categoría de las teorías

Let $G = \langle V, E \rangle$ a graph and $\delta : G \to \mathsf{Th}_0$ a finite diagram in Th_0. Let $T_v = \langle \langle \{f_i\}_{i \in \mathcal{I}_v} \rangle, \Gamma_v \rangle \in |\mathsf{Th}_0|$ for all $v \in V$. Then, $\langle \langle \Sigma, \Gamma \rangle, \{\gamma_v : T_v \to \langle \Sigma, \Gamma \rangle\}_{v \in V} \rangle$ where $\Sigma = \langle \left(\bigcup_{v \in V} \{f_i\}_{i \in \mathcal{I}_v} \right) | \overset{*}{\longleftrightarrow}^{\delta} \rangle$, $\Gamma = \bigcup_{v \in V} \mathbf{Sen}(\gamma_v)(\Gamma_v)$, and $\gamma_v(f_i) = [\![f_i]\!]_{\overset{*}{\longleftrightarrow}^{\delta}}$ for all $i \in \mathcal{I}_v$, is a co-limit of δ.

Propiedades emergentes

Secuencias con longitud

$$\langle\langle seq, bool, nat, [], \&\&, esVacía?, prim, resto, long, 0, suc\rangle, Ax_{long}\rangle$$

Secuencias con reverso

$$\langle\langle seq, bool, [], \&\&, esVacía?, prim, resto, reverso\rangle, Ax_{rev}\rangle$$

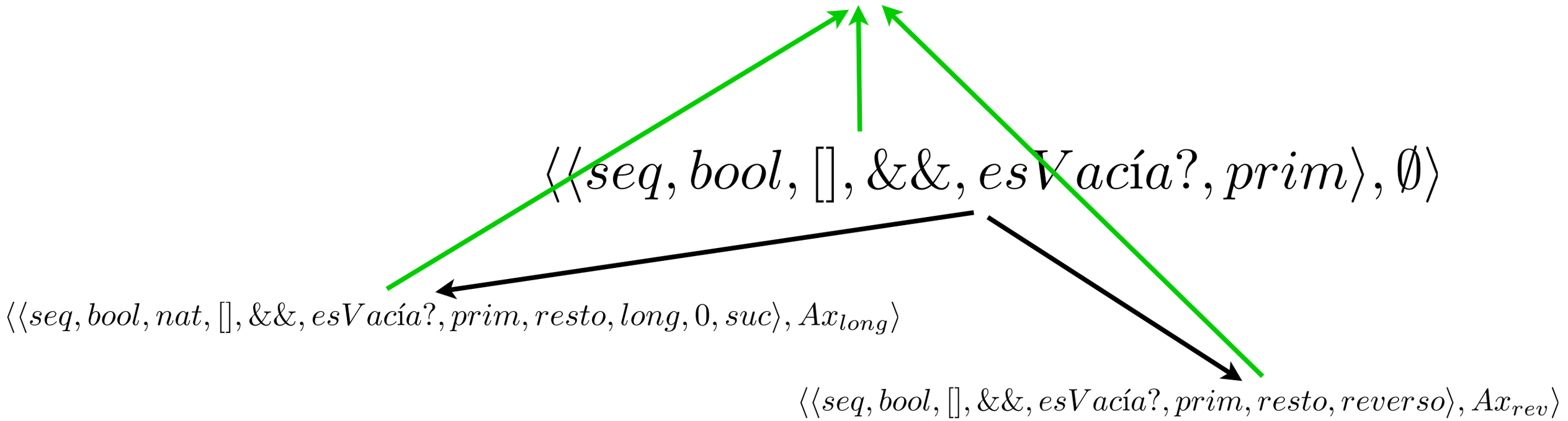
Propiedad sobre secuencias

$$(\forall s : seq)\ long(s) = long(reverso(s))$$

La propiedad **no** se sigue de ninguna de las dos especificaciones, pero...

Propiedades emergentes

$\langle\langle seq, bool, [], \&\&, esVacía?, prim, resto, long, 0, suc, reverso\rangle, Ax_{long} \cup Ax_{rev}\rangle$



La propiedad **sí** se sigue del vértice del co-límite del diagrama formado de las dos especificaciones y el channel apropiado.

Resumen

Hasta aquí hemos definido formalmente el mecanismo de composición de especificaciones lógicas para el caso homogéneo. Algunas propiedades:

- esta forma de componer especificaciones no sólo preserva las propiedades de las partes sino que hace emerger propiedades nuevas para la composición,
- la especificación resultante podría ser contradictoria.

A partir de mañana analizaremos cómo componer especificaciones para el caso heterogéneo.

Especificaciones heterogéneas

Carlos G. Lopez Pombo
Departamento de computación, FCEyN, UBA
y CONICET

Panorama general

Las especificaciones heterogéneas implican:

- la presencia de una multiplicidad de lógicas usadas para describir partes no necesariamente disjunta de un mismo sistema,
- la necesidad de combinar las vistas parciales de un sistema en una descripción completa.

¿Dónde estamos?

“There is a population explosion among the logical systems used in computing science. Examples include first-order logic, equational logic, Horn-clause logic, higher-order logic, infinitary logic, dynamic logic. intuitionistic logic, order-sorted logic, and temporal logic; moreover, there is a tendency for each theorem prover to have its own idiosyncratic logical system.”

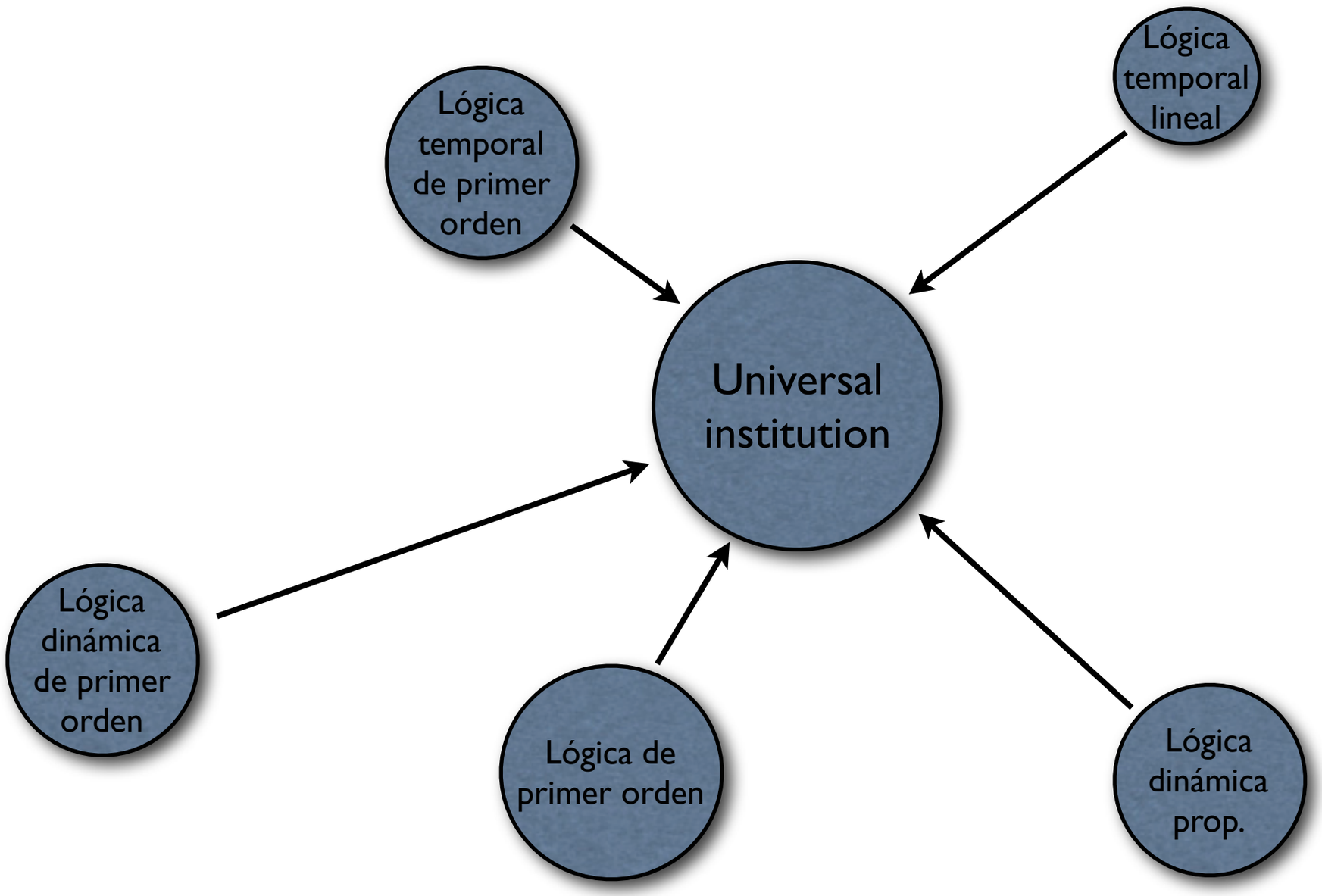
[GB84] Goguen & Burstall, “Introducing Institutions”, 1984

¿A dónde vamos?

"... we should strive at a development of a convenient to use proof theory with support tools for a sufficiently rich "universal institution" and then reuse it for other institutions linked to it by institution representations."

[Tar97] Andrzej Tarlecki, "Moving between logical systems", 1998

La propuesta



La propuesta

Algunas preguntas que surgen de esto:

- si voy a "traducir" todo a un mismo lenguaje, ¿por qué no escribir la especificación directamente en él?
- ¿qué propiedades debe satisfacer una de estas "flechas" (*representation maps*)?
- ¿los *representation maps* son la única alternativa?¿de haber otras, cómo elegir?

Un poco más...

Transformación natural [Fia05]

Let $\mathsf{C}, \mathsf{D} \in \mathsf{Cat}$, $\psi : \mathsf{C} \to \mathsf{D}$ and $\phi : \mathsf{C} \to \mathsf{D}$ be functors, then $\tau : \psi \to \phi$ is a *natural transformation* if it is a function that assigns to each object $c \in |\mathsf{C}|$ a morphism $\tau_c : \psi(c) \to \phi(c)$ such that for each $c, c' \in |\mathsf{C}|$, $f : c \to c'$ morphism in C, the following diagram commutes:

$$\begin{array}{ccc} \psi(c) & \xrightarrow{\tau_c} & \phi(c) \\ \downarrow{\scriptstyle \psi(f)} & & \downarrow{\scriptstyle \phi(f)} \\ \psi(c') & \xrightarrow{\tau_{c'}} & \phi(c') \end{array}$$

Un poco más...

Morfismo entre instituciones [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

$\gamma^{Sign} : \mathsf{Sign}' \to \mathsf{Sign}$ is a functor,

Un poco más...

Morfismo entre instituciones [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

$\gamma^{Sen} : \gamma^{Sign} \circ \mathbf{Sen} \to \mathbf{Sen}'$, is a natural transformation (i.e. a natural family of functions $\gamma^{Sen}_{\Sigma'} : \mathbf{Sen}(\gamma^{Sign}(\Sigma')) \to \mathbf{Sen}'(\Sigma'))$, such that for each $\Sigma'_1, \Sigma'_2 \in |\mathsf{Sign}'|$ and $\sigma' : \Sigma'_1 \to \Sigma'_2$ morphism in Sign',

$$\begin{array}{ccc}
\mathbf{Sen}(\gamma^{Sign}(\Sigma'_2)) & \xrightarrow{\gamma^{Sen}_{\Sigma'_2}} & \mathbf{Sen}'(\Sigma'_2) \\
\uparrow{\scriptstyle \mathbf{Sen}(\gamma^{Sign}(\sigma'))} & & \uparrow{\scriptstyle \mathbf{Sen}'(\sigma')} \\
\mathbf{Sen}(\gamma^{Sign}(\Sigma'_1)) & \xrightarrow{\gamma^{Sen}_{\Sigma'_1}} & \mathbf{Sen}'(\Sigma'_1)
\end{array}
\qquad
\begin{array}{c}
\Sigma'_2 \\
\uparrow{\scriptstyle \sigma'} \\
\Sigma'_1
\end{array}$$

Un poco más...

Morfismo entre instituciones [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

$\gamma^{Mod} : \mathbf{Mod}' \to (\gamma^{Sign})^{\mathsf{op}} \circ \mathbf{Mod}$, is a natural transformation (i.e. the family of functors $\gamma^{Mod}_{\Sigma'} : \mathbf{Mod}'(\Sigma') \to \mathbf{Mod}((\gamma^{Sign})^{\mathsf{op}}(\Sigma'))$ is natural), such that for each $\Sigma'_1, \Sigma'_2 \in |\mathsf{Sign}'|$ and $\sigma' : \Sigma'_1 \to \Sigma'_2$ morphism in Sign',

$$\begin{array}{ccc}
\mathbf{Mod}'(\Sigma'_2) & \xrightarrow{\gamma^{Mod}_{\Sigma'_2}} & \mathbf{Mod}((\gamma^{Sign})^{\mathsf{op}}(\Sigma'_2)) \\
\downarrow{\scriptstyle \mathbf{Mod}'(\sigma'^{\mathsf{op}})} & & \downarrow{\scriptstyle \mathbf{Mod}((\gamma^{Sign})^{\mathsf{op}}(\sigma'^{\mathsf{op}}))} \\
\mathbf{Mod}'(\Sigma'_1) & \xrightarrow{\gamma^{Mod}_{\Sigma'_1}} & \mathbf{Mod}((\gamma^{Sign})^{\mathsf{op}}(\Sigma'_1))
\end{array}
\qquad
\begin{array}{c}
\Sigma'_2 \\
\uparrow{\scriptstyle \sigma'} \\
\Sigma'_1
\end{array}$$

The functor $(\gamma^{Sign})^{\mathsf{op}} : \mathsf{Sign}'^{\mathsf{op}} \to \mathsf{Sign}^{\mathsf{op}}$ is the same as $\gamma^{Sign} : \mathsf{Sign}' \to \mathsf{Sign}$ but considered between the opposite categories.

Un poco más...

Morfismo entre instituciones [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

...

such that for any $\Sigma' \in |\mathsf{Sign}'|$, the function $\gamma^{Sen}_{\Sigma'} : \mathbf{Sen}(\gamma^{Sign}(\Sigma')) \to \mathbf{Sen}'(\Sigma')$ and the functor $\gamma^{Mod}_{\Sigma'} : \mathbf{Mod}'(\Sigma') \to \mathbf{Mod}((\gamma^{Sign})^{\mathsf{op}}(\Sigma'))$ preserves the following satisfaction condition: for any $\alpha \in \mathbf{Sen}(\gamma^{Sign}(\Sigma'))$ and $\mathcal{M} \in |\mathbf{Mod}(\Sigma')|$,

$$\mathcal{M} \models_{\Sigma'} \gamma^{Sen}_{\Sigma'}(\alpha) \text{ iff } \gamma^{Mod}_{\Sigma'}(\mathcal{M}) \models_{\gamma^{Sign}(\Sigma')} \alpha \ .$$

Morfismos entre instituciones

Los morfismos entre instituciones capturan cómo una institución "más rica" (rica respecto de su teoría de modelos) se construye sobre otras "más pobres".

Las instituciones, junto con los morfismos entre instituciones forman una categoría (Ins). Ins da soporte para la construcción incremental de especificaciones a través de la construcción de los límites correspondientes.

Morfismos entre instituciones

Preservación de consecuencia [Tar96]

Semantic consequence is preserved by institution morphism: for any institution morphism $\mu : \mathsf{I}' \to \mathsf{I}$, signatures $\Sigma \in |\mathsf{Sign}|$ and $\Sigma' \in |\mathsf{Sign}'|$ such that $\mu^{Sign}(\Sigma') = \Sigma$, set $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ of Σ-sentences, and Σ-sentence $\phi \in \mathbf{Sen}(\Sigma)$,

if $\Gamma \models^{\mathsf{I}}_{\Sigma} \phi$, then $\mu^{Sen}_{\Sigma'}(\Gamma) \models^{\mathsf{I}'}_{\Sigma'} \mu^{Sen}_{\Sigma'}(\phi)$.

Reflejo de consecuencia [Tar96]

Semantic consequence is reflected by institution morphism: for any institution morphism $\mu : \mathsf{I}' \to \mathsf{I}$, signatures $\Sigma \in |\mathsf{Sign}|$ and $\Sigma' \in |\mathsf{Sign}'|$ such that $\mu^{Sign}(\Sigma') = \Sigma$, set $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ of Σ-sentences, and Σ-sentence $\phi \in \mathbf{Sen}(\Sigma)$, such that each model $\mathcal{M} \in |\mathbf{Mod}(\langle \Sigma, \Gamma \rangle)|$ has a μ-expansion to a Σ'-model (i.e. there exists $\mathcal{M}' \in |\mathbf{Mod}'(\Sigma')|$ such that $\mu^{Mod}_{\Sigma'}(\mathcal{M}') = \mathcal{M}$)

$\Gamma \models^{\mathsf{I}}_{\Sigma} \phi$, if and only if $\mu^{Sen}_{\Sigma}(\Gamma) \models^{\mathsf{I}'}_{\Sigma'} \mu^{Sen}_{\Sigma}(\phi)$.

Un poco más...

***Representation maps* entre instituciones** [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

$\gamma^{Sign} : \mathsf{Sign} \to \mathsf{Sign}'$ is a functor,

Un poco más...

Representation maps **entre instituciones** [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

$\gamma^{Sen} : \mathbf{Sen} \to \gamma^{Sign} \circ \mathbf{Sen}'$, is a natural transformation (i.e. a natural family of functions $\gamma^{Sen}_\Sigma : \mathbf{Sen}(\Sigma) \to \mathbf{Sen}'(\gamma^{Sign}(\Sigma)))$, such that for each $\Sigma_1, \Sigma_2 \in |\mathsf{Sign}|$ and $\sigma : \Sigma_1 \to \Sigma_2$ morphism is Sign,

$$
\begin{array}{ccc}
\mathbf{Sen}(\Sigma_2) & \xrightarrow{\gamma^{Sen}_{\Sigma_2}} & \mathbf{Sen}'(\gamma^{Sign}(\Sigma_2)) \\
\uparrow{\scriptstyle \mathbf{Sen}(\sigma)} & & \uparrow{\scriptstyle \mathbf{Sen}'(\gamma^{Sign}(\sigma))} \\
\mathbf{Sen}(\Sigma_1) & \xrightarrow{\gamma^{Sen}_{\Sigma_1}} & \mathbf{Sen}'(\gamma^{Sign}(\Sigma_1))
\end{array}
\qquad
\begin{array}{c}
\Sigma_2 \\
\uparrow{\scriptstyle \sigma} \\
\Sigma_1
\end{array}
$$

Un poco más...

***Representation maps* entre instituciones** [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

$\gamma^{Mod} : (\gamma^{Sign})^{\mathsf{op}} \circ \mathbf{Mod}' \to \mathbf{Mod}$, is a natural transformation (i.e. the family of functors $\gamma^{Mod}_\Sigma : \mathbf{Mod}'((\gamma^{Sign})^{\mathsf{op}}(\Sigma)) \to \mathbf{Mod}(\Sigma)$ is natural), such that for each $\Sigma_1, \Sigma_2 \in |\mathsf{Sign}|$ and $\sigma : \Sigma_1 \to \Sigma_2$ morphism in Sign,

$$\begin{array}{ccc}
\mathbf{Mod}'((\gamma^{Sign})^{\mathsf{op}}(\Sigma_2)) & \xrightarrow{\gamma^{Mod}_{\Sigma_2}} & \mathbf{Mod}(\Sigma_2) \\
\Big\downarrow{\scriptstyle \mathbf{Mod}'((\gamma^{Sign})^{\mathsf{op}}(\sigma^{)})} & & \Big\downarrow{\scriptstyle \mathbf{Mod}(\sigma^{\mathsf{op}})} \\
\mathbf{Mod}'((\gamma^{Sign})^{\mathsf{op}}(\Sigma_1)) & \xrightarrow{\gamma^{Mod}_{\Sigma_1}} & \mathbf{Mod}(\Sigma_1)
\end{array}
\qquad
\begin{array}{c}
\Sigma_2 \\
\Big\uparrow{\scriptstyle \sigma} \\
\Sigma_1
\end{array}$$

The functor $(\gamma^{Sign})^{\mathsf{op}} : \mathsf{Sign}'^{\mathsf{op}} \to \mathsf{Sign}^{\mathsf{op}}$ is the same as $\gamma^{Sign} : \mathsf{Sign}' \to \mathsf{Sign}$ but considered between the opposite categories.

Un poco más...

***Representation maps* entre instituciones** [Tar96]

Let $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in |\mathsf{Sign}|}\rangle$ and $\langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}'|}\rangle$ be the institutions I and I' respectively, then $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod}\rangle : I \to I'$ is an *institution morphism* if and only if:

...

such that for any $\Sigma \in |\mathsf{Sign}|$, the function $\gamma^{Sen}_\Sigma : \mathbf{Sen}(\Sigma) \to \mathbf{Sen}'(\gamma^{Sign}(\Sigma))$ and the functor $\gamma^{Mod}_\Sigma : \mathbf{Mod}'((\gamma^{Sign})^{\mathsf{op}}(\Sigma)) \to \mathbf{Mod}(\Sigma)$ preserves the following satisfaction condition: for any $\alpha \in \mathbf{Sen}(\Sigma)$ and $\mathcal{M}' \in |\mathbf{Mod}(\gamma^{Sign}(\Sigma))|$,

$$\mathcal{M}' \models_{(\gamma^{Sign})^{\mathsf{op}}(\Sigma)} \gamma^{Sen}_\Sigma(\alpha) \;\text{ iff }\; \gamma^{Mod}_\Sigma(\mathcal{M}') \models_\Sigma \alpha \ .$$

Representation maps entre instituciones

Los *representation maps* entre instituciones capturan cómo una institución "más pobre" (pobre respecto de su teoría de modelos) puede ser interpretada en otras "más ricas".

Las instituciones, junto con los *representation maps* entre instituciones forman una categoría (R-Ins). R-Ins da soporte para la unificación de especificaciones a través de la construcción de los co-límites correspondientes.

Representation maps entre instituciones

Preservación de consecuencia [Tar96]

Semantic consequence is preserved by institution *representation maps*: for any institution *representation map* $\mu : \mathsf{I} \to \mathsf{I}'$, signature $\Sigma \in |\mathsf{Sign}|$, set $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ of Σ-sentences, and Σ-sentence $\phi \in \mathbf{Sen}(\Sigma)$,

$$\text{if } \Gamma \models^{\mathsf{I}}_{\Sigma} \phi, \text{ then } \mu^{Sen}_{\Sigma}(\Gamma) \models^{\mathsf{I}'}_{\mu^{Sign}(\Sigma)} \mu^{Sen}_{\Sigma}(\phi).$$

Representation maps entre instituciones

Preservación de consecuencia [Tar96]

La definición de *representation map* fuerza la preservación de la consecuencia semántica. Sin embargo, podrían existir modelos en la institución I que no tengan un representante en la institución I'.

Luego, toda propiedad universalmente válida en la institución I', necesariamente será universalmente válida en la institución I. Es decir que si existiera un cálculo completo para I', entonces tendríamos un cálculo completo para I.

Representation maps entre instituciones

Reflejo de consecuencia [Tar96]

Semantic consequence is reflected by institution *representation maps*: for any institution *representation map* $\mu : \mathsf{I} \to \mathsf{I}'$, signature $\Sigma \in |\mathsf{Sign}|$, set $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ of Σ-sentences, and Σ-sentence $\phi \in \mathbf{Sen}(\Sigma)$, such that each model $\mathcal{M} \in| \mathbf{Mod}(\langle \Sigma, \Gamma \rangle)|$ has a μ-expansion to a $\mu^{Sign}(\Sigma)$-model (i.e. there exists $\mathcal{M}' \in |\mathbf{Mod}'(\mu^{Sign}(\Sigma))|$ such that $\mu_{\Sigma}^{Mod}(\mathcal{M}') = \mathcal{M}$)

$$\Gamma \models^{\mathsf{I}}_{\Sigma} \phi, \text{ if and only if } \mu_{\Sigma}^{Sen}(\Gamma) \models^{\mathsf{I}'}_{\mu^{Sign}(\Sigma)} \mu_{\Sigma}^{Sen}(\phi).$$

Representation maps entre instituciones

Reflejo de consecuencia [Tar96]

La propiedad de extensibilidad de los modelos de I a los modelos de I' fuerza el reflejo de la consecuencia semántica. De esta forma, sabemos que todo modelo de I' puede ser "traducido" a un modelo de I.

Luego, si una propiedad tiene un contraejemplo en I', necesariamente existirá un contraejemplo en I.

Representation maps entre instituciones

Ejemplo (lógica ecuacional en lógica de primer orden)

Representation maps entre instituciones

Ejemplo (lógica ecuacional en lógica de primer orden)

- La signaturas ecuacionales (sólo posee símbolos de función), son traducidas a signaturas de primer orden en las que el conjunto de predicados es vacío.

Representation maps entre instituciones

Ejemplo (lógica ecuacional en lógica de primer orden)

- La signaturas ecuacionales (sólo posee símbolos de función), son traducidas a signaturas de primer orden en las que el conjunto de predicados es vacío.
- Las sentencias son exáctamente las mismas puesto que las ecuaciones son fórmulas bien formadas de primer orden.

Representation maps entre instituciones

Ejemplo (lógica ecuacional en lógica de primer orden)

- La signaturas ecuacionales (sólo posee símbolos de función), son traducidas a signaturas de primer orden en las que el conjunto de predicados es vacío.
- Las sentencias son exáctamente las mismas puesto que las ecuaciones son fórmulas bien formadas de primer orden.
- Los modelos ecuacionales se obtienen tomando el reducto correspondiente a los símbolos de función.

Representation maps entre instituciones

Ejemplo (lógica ecuacional en lógica de primer orden)

- La signaturas ecuacionales (sólo posee símbolos de función), son traducidas a signaturas de primer orden en las que el conjunto de predicados es vacío.
- Las sentencias son exáctamente las mismas puesto que las ecuaciones son fórmulas bien formadas de primer orden.
- Los modelos ecuacionales se obtienen tomando el reducto correspondiente a los símbolos de función.

¿Se satisface la propiedad que extensibilidad de los modelos ecuacionales a modelos de primer orden?

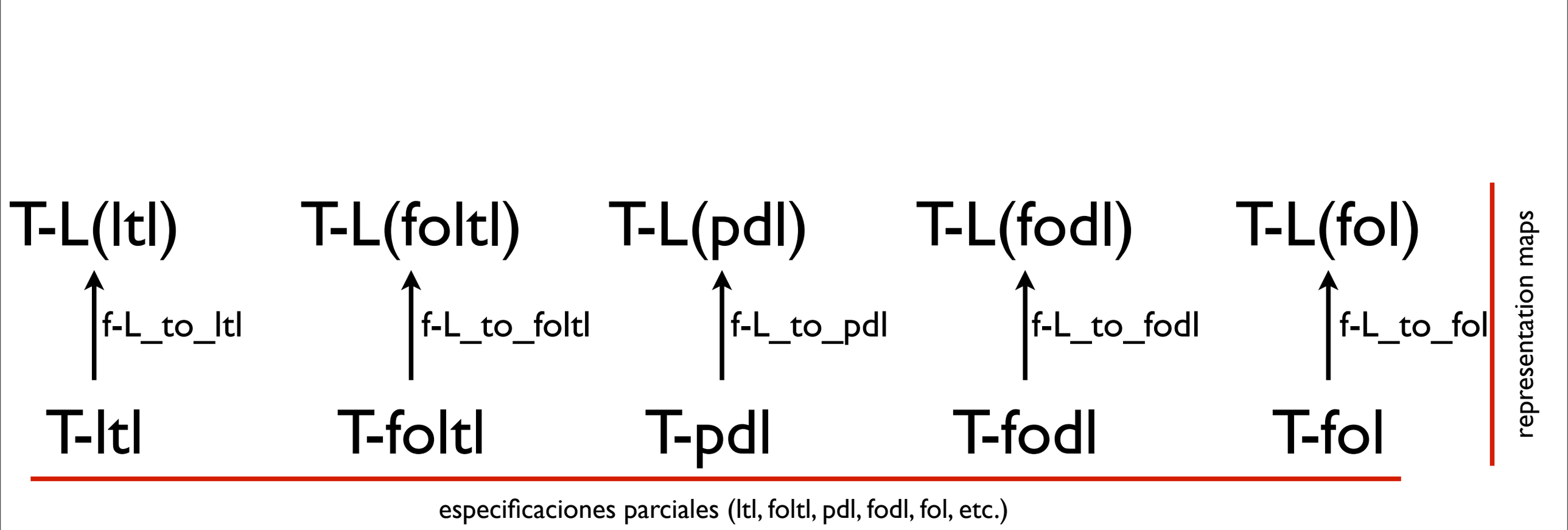
La propuesta (más concreta)...

La propuesta (más concreta)...

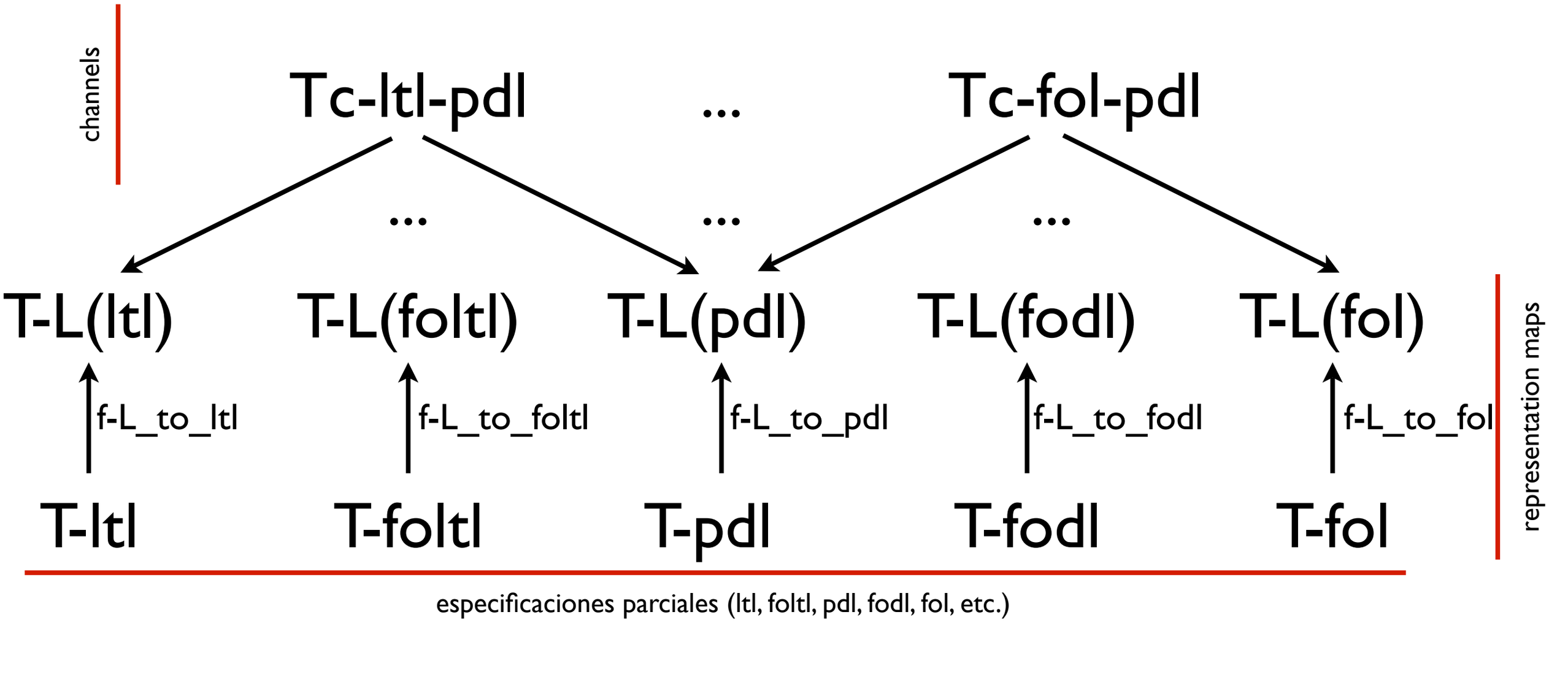
T-ltl T-foltl T-pdl T-fodl T-fol

especificaciones parciales (ltl, foltl, pdl, fodl, fol, etc.)

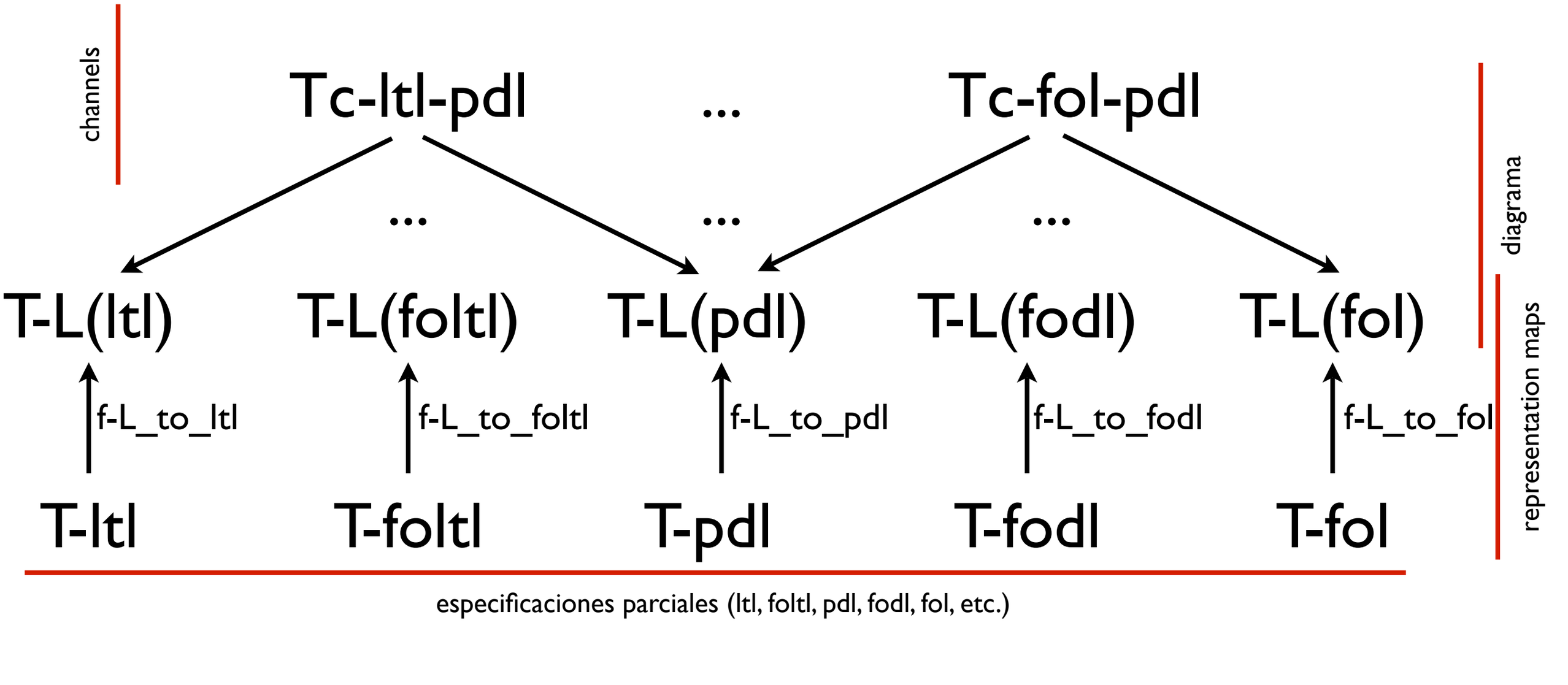
La propuesta (más concreta)...



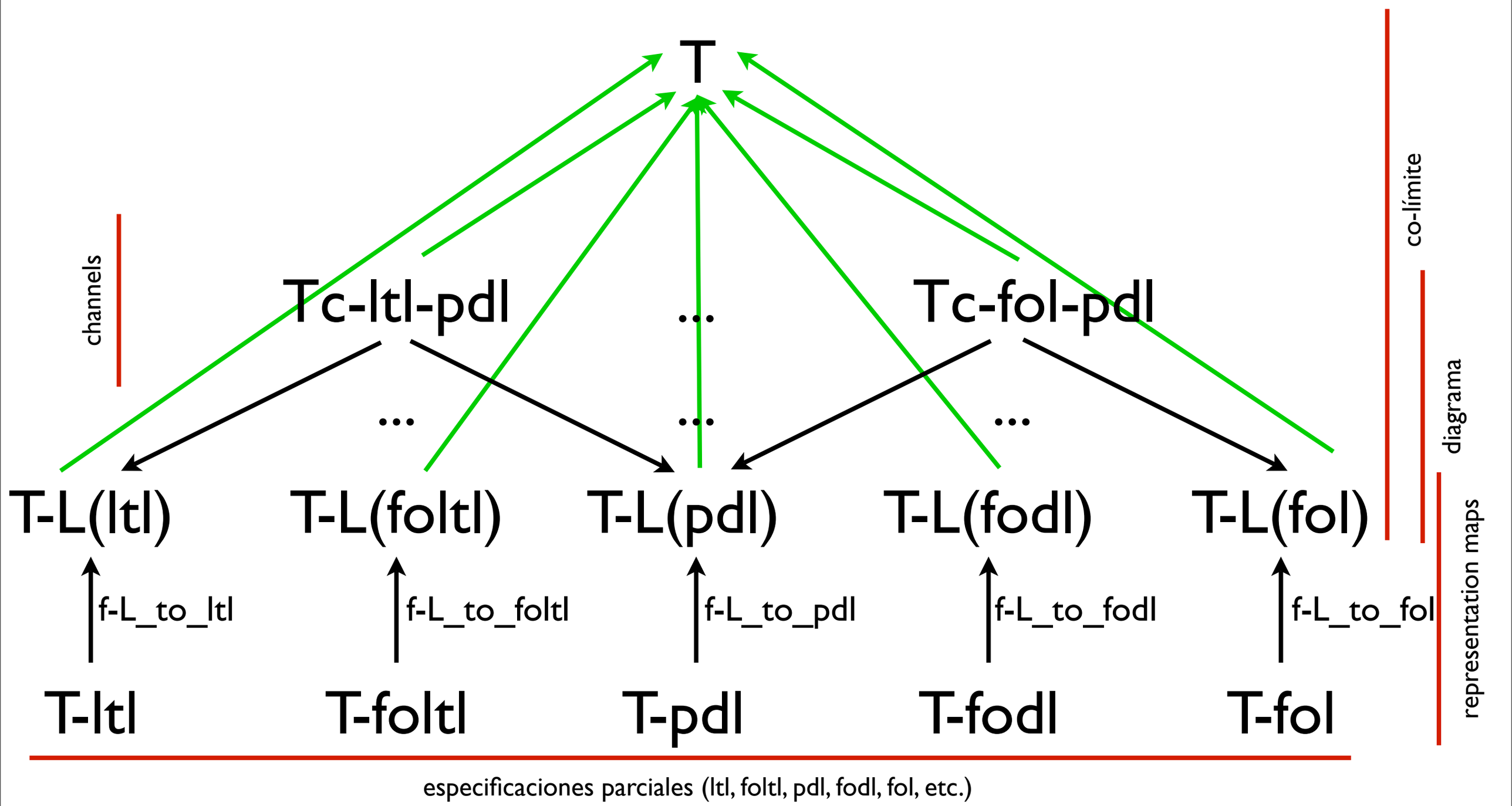
La propuesta (más concreta)...



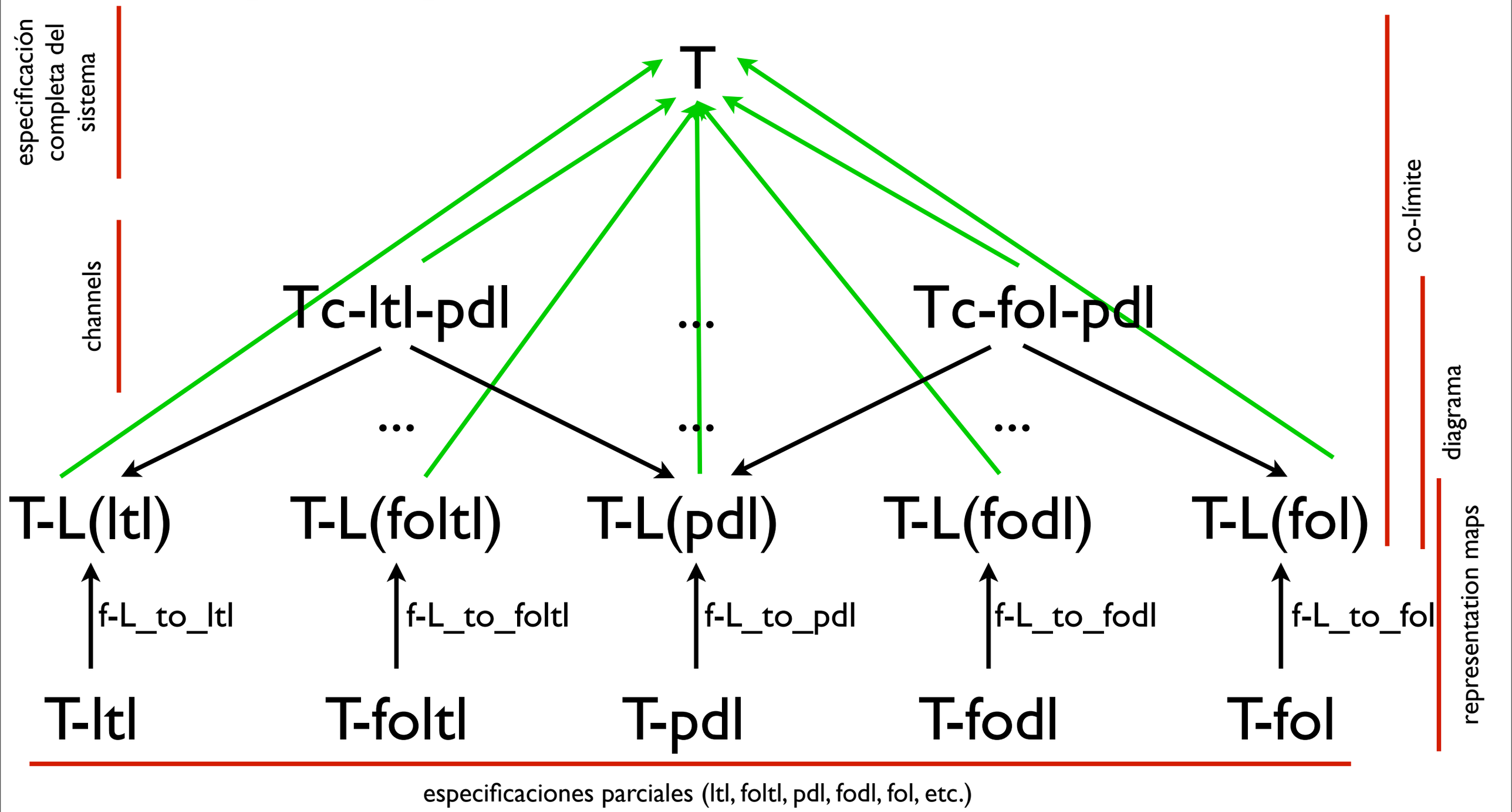
La propuesta (más concreta)...



La propuesta (más concreta)...



La propuesta (más concreta)...



Representation maps entre instituciones

Preservación y reflejo de consecuencia

En definitiva, estas propiedades nos proporcionan la posibilidad de reemplazar todo razonamiento sobre especificaciones escritas en la institución I por razonamiento en la institución I'.

Luego, basta con encontrar una institución suficientemente poderosa, para la que se puedan desarrollar herramientas de validación y/o verificación, y a partir del uso de *representation maps* aplicar estas técnicas.

Representation maps entre instituciones

Un detalle pendiente [Mes89]

El único cabo suelto que ha quedado es que nuestra definición de *representation map* funciona para signaturas pero para implementar nuestra propuesta debemos ser capaces de aplicar los *representation maps* sobre teorías (*maps of institutions*).

Esto se obtiene a partir de extender la definición de *representation map* a la categoría Th_0, en lugar de Sign, solicitando que $\gamma^{Th_0} : \mathsf{Th}_0{}^{\mathsf{I}} \to \mathsf{Th}_0{}^{\mathsf{I}'}$ sea $\gamma^{Sen} - sensible$.

Es decir que las propiedades demostrables en $\gamma^{Th_0}(\langle \Sigma, \Gamma \rangle)$ sólo dependan de $\gamma^{Th_0}(\langle \Sigma, \emptyset \rangle)$ y $\gamma^{Sen}(\Gamma)$.

$$\begin{aligned} \gamma^{Th_0}(\langle \Sigma, \Gamma \rangle) &= \langle \Sigma', \Gamma' \cup \gamma^{Sen}(\Gamma), \text{ tal que:} \\ \langle \Sigma', \Gamma' \rangle &= \gamma^{Th_0}(\langle \Sigma', \emptyset \rangle) \end{aligned}$$

Fork Algebras

[Fri02]

- they are isomorphic to algebras whose domain is a set of binary relations (Frias et al. in [**FBH97**] and Gyuris in [**Gyu97**]),
- finite equational calculus (Frias et al. in [**FHV97**]),
- expressive power capable of providing an interpretation language for many logics. Given a logic $\mathcal{L}$, an interpretation is a relational algebraization of $\mathcal{L}$. This is done by resorting to a semantics-preserving mapping $T_{\mathcal{L}} : \mathit{Formulas}_{\mathcal{L}} \rightarrow \mathit{RelDes}(X)$ for some set of relational variables X, translating $\mathcal{L}$-formulas to relational terms. Let $\Gamma \cup \{\alpha\} \subseteq \mathit{Formulas}_{\mathcal{L}}$, then the property of being semantic-preserving is characterized by the following condition:

$$\Gamma \models_{\mathcal{L}} \alpha \iff T_{\mathcal{L}}(\gamma) = \gamma \in \Gamma \vdash_{\mathsf{CFAU}} T_{\mathcal{L}}(\alpha) = 1 :$$

  - interpretability of first-order predicate logic (FOL) in fork algebra [**Fri02**],
  - interpretability of PDL in fork algebra [**FO98**],
  - interpretability of first-order dynamic logic (FODL) in fork algebra [**FBM02**],
  - interpretability of LTL, TL [**FL03**] and their first-order versions in fork algebra [**FL06**],
  - interpretability of propositional dynamic linear temporal logic (DLTL) in fork algebra [**FGLR05**].

Fork Algebras

[Fri02]

Términos

Sea $F = \{f_i\}_{i \in \mathcal{I}}$ un conjunto de símbolos de función, $V = \{v_k\}_{k \in \mathcal{K}}$. Luego, $RelDes$ es el conjunto más chico que satisface T que satisface:

- $V \subseteq T$
- $1, 0, 1' \subseteq T$
- si $t_1, \ldots, t_n \in T$, entonces $f_i(t_1, \ldots, t_n) \in T$
- si $t_1 \in T$, entonces $\bar{t}, \breve{t}, t^*, t^\diamond \in T$
- si $t_1, t_2 \in T$, entonces $t_1 + t_2, t_1 \cdot t_2, t_1 \; ; \; t_2, t_1 \nabla t_2 \in T$

Fórmulas

- $t_1 = t_2$, si $t_1, t_2 \in T$

Fork Algebras

[Fri02]

Modelos

Sea U un conjunto de elementos (urelementos), se define A de la siguiente forma:

- $U \subseteq A$
- si, $u_1, u_2 \in A$, entonces $u_1 \star u_2 \in A$

Fork Algebras

[Fri02]

Modelos

Sea U un conjunto de elementos (urelementos). Una *full proper closure fork algebra con urelementos* es:

$$\langle 2^{A\times A}, A\times A, \cup, \emptyset, \cap, -, Id_{A\times A}, \circ, {}^T, \nabla, {}^*, {}^\diamond\rangle$$

tal que,

$$\begin{aligned}
R\circ S &= \{\langle a,b\rangle \mid \text{existe } c\in A \text{ tal que } \langle a,c\rangle\in R \text{ y } \langle c,b\rangle\in S\}\\
R^T &= \{\langle a,b\rangle \mid \langle b,a\rangle\in R\}\\
R\nabla S &= \{\langle a, b\star c\rangle \mid \langle a,b\rangle\in R \text{ y } \langle a,c\rangle\in S\}\\
R^* &= \bigcup_{i<\omega} R^i\\
R^\diamond &\in R
\end{aligned}$$

Fork Algebras

[Fri02]

Modelos

Sea $F = \{f_i\}_{i \in \mathcal{I}}$ un conjunto de símbolos de función. Un modelo para la signatura $\langle F \rangle$ es una estructura $\langle P, \{\underline{f_i}\}_{i \in \mathcal{I}} \rangle$ tal que:

- P es una full proper closure fork algebra con urelementos,
- $\underline{f_i} : P^{arity(f_i)} \to P,$ para todo $i \in \mathcal{I}$.

Fork Algebras

[Fri02]

Semántica

Sea $F = \{f_i\}_{i \in \mathcal{I}}$ un conjunto de símbolos de función, $V = \{v_k\}_{k \in \mathcal{K}}$. Sea $val : V \to 2^{A \times A}$ una valuación de las variables de V en $2^{A \times A}$. Se define $m_{val} : RelDes \to 2^{A \times A}$ de la siguiente forma:

$$
\begin{aligned}
m_{val}(v) &= val(v), \text{para toda } v \in V. \\
m_{val}(f_i(t_1, \ldots, t_n)) &= \underline{f_i}(m_{val}(t_1), \ldots, m_{val}(t_n)), \text{para toda } i \in \mathcal{I}. \\
m_{val}(1) &= A \times A \\
m_{val}(t_1 + t_2) &= m_{val}(t_1) \cup m_{val}(t_2) \\
m_{val}(0) &= \emptyset \\
m_{val}(t_1 \cdot t_2) &= m_{val}(t_1) \cap m_{val}(t_2) \\
m_{val}(\overline{t_1}) &= \overline{m_{val}(t_1)} \\
m_{val}(1') &= Id_{A \times A} \\
m_{val}(t_1 \ ; \ t_2) &= m_{val}(t_1) \circ m_{val}(t_2) \\
m_{val}(t_1 \nabla t_2) &= m_{val}(t_1) \nabla m_{val}(t_2) \\
m_{val}({t_1}^*) &= m_{val}(t_1)^* \\
m_{val}({t_1}^\diamond) &= m_{val}(t_1)^\diamond
\end{aligned}
$$

Fork Algebras

[Fri02]

Semántica

Sea $F = \{f_i\}_{i \in \mathcal{I}}$ un conjunto de símbolos de función, $V = \{v_k\}_{k \in \mathcal{K}}$. Sea $val : V \to 2^{A \times A}$ una valuación de las variables de V en $2^{A \times A}$. Se $t_1 = t_2$ vale si y sólo si $m_{val}(t_1) = m_{val}(t_2)$.

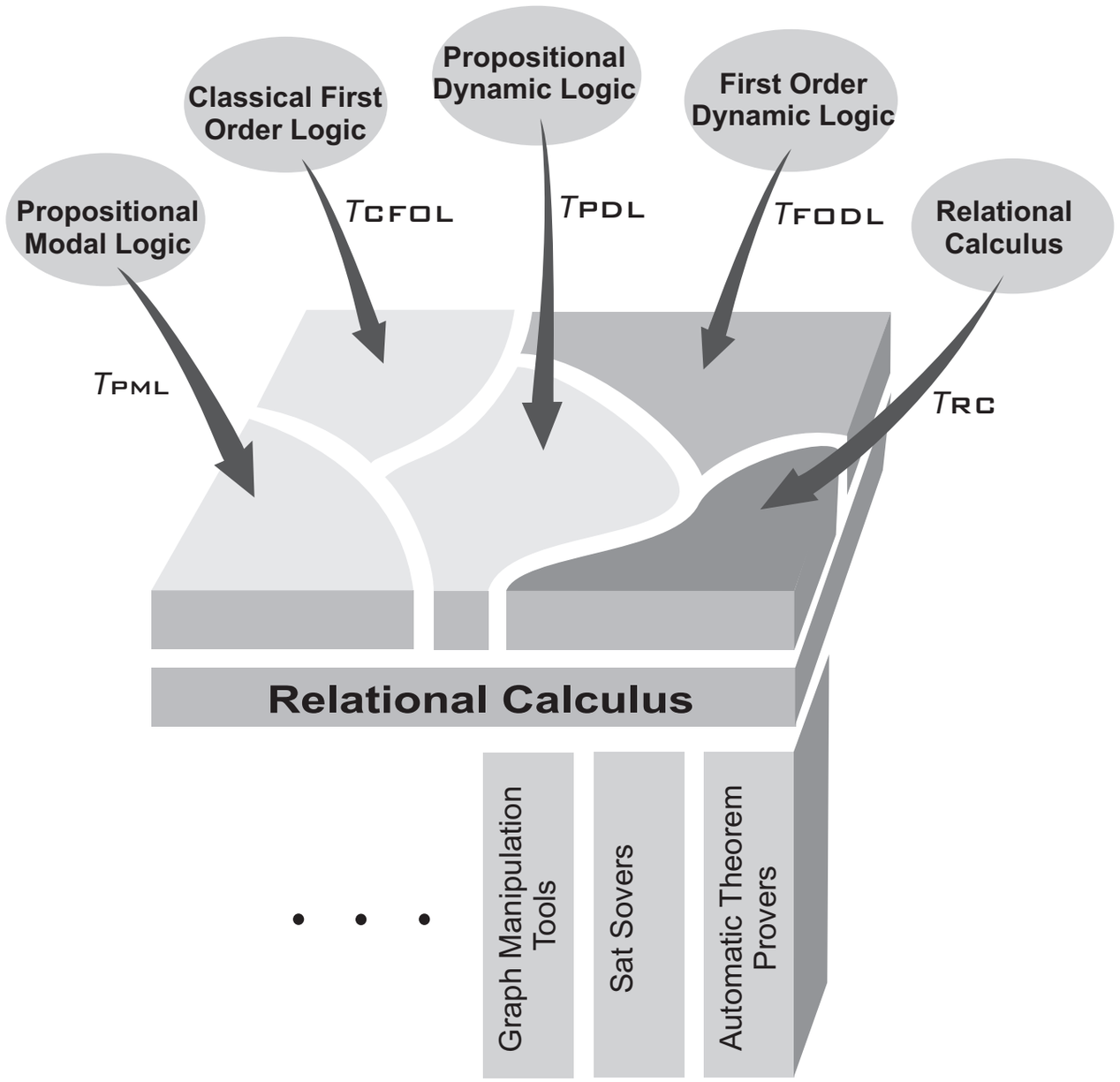
Representation maps entre instituciones [FL03]

Ejemplo (lógica lineal proposicional en fork algebra) [LF06]

¡Pizarrón!

$\mathbf{Ar}_{\mathbf{g}}entum$

[Lop07]



Especificaciones estructuradas

Carlos G. Lopez Pombo
Departamento de computación, FCEyN, UBA
y CONICET

Motivación

Hasta aquí hemos aprendido cómo a partir de teorías lógicas que formalizan aspectos diferentes de un sistema podemos obtener una descripción completa del mismo.

Las teorías con las que hemos trabajado hasta ahora carecen de estructura, son sólo un conjunto de axiomas caracterizando un comportamiento determinado.

Normalmente las especificaciones (que en nuestro caso son teorías) se escriben a partir de sucesivos refinamientos hasta llegar a una que nos satisfaga como formalización del aspecto que está siendo descripto.

Motivación

Contar con operaciones que permitan estructurar especificaciones aporta:

- **fundamentos** para comprender la práctica habitual en el desarrollo de software,
- **claridad** en la lectura de una especificación puesto que se tiene el rastro de cómo fue construida,
- **modularidad** para facilitar el reuso de especificaciones,
- **facilidad** para demostrar propiedades del sistema.

Entailment system

[Mes89]

An *entailment system* is a structure of the form $\langle \mathsf{Sign}, \mathbf{Sen}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ satisfying the following conditions:

- Sign is a category of signatures,
- $\mathbf{Sen} : \mathsf{Sign} \to \mathsf{Set}$ is a functor (let $\Sigma \in |\mathsf{Sign}|$, then $\mathbf{Sen}(\Sigma)$ returns the set of Σ-sentences),
- $\{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}$, where $\vdash^{\Sigma} \subseteq 2^{\mathbf{Sen}(\Sigma)} \times \mathbf{Sen}(\Sigma)$, is a family of binary relations such that for any $\Sigma, \Sigma' \in |\mathsf{Sign}|$, $\{\phi\} \cup \{\phi_i\}_{i \in \mathcal{I}} \subseteq \mathbf{Sen}(\Sigma)$, $\Gamma, \Gamma' \subseteq \mathbf{Sen}(\Sigma)$ the following conditions are satisfied:
  1. reflexivity: $\{\phi\} \vdash^{\Sigma} \phi$,
  2. monotonicity: if $\Gamma \vdash^{\Sigma} \phi$ and $\Gamma \subseteq \Gamma'$, then $\Gamma' \vdash^{\Sigma} \phi$,
  3. transitivity: if $\Gamma \vdash^{\Sigma} \phi_i$ for all $i \in \mathcal{I}$ and $\{\phi_i\}_{i \in \mathcal{I}} \vdash^{\Sigma} \phi$, then $\Gamma \vdash^{\Sigma} \phi$, and
  4. $\vdash$-translation: if $\Gamma \vdash^{\Sigma} \phi$, then for any morphism $\sigma : \Sigma \to \Sigma'$ in Sign, $\mathbf{Sen}(\sigma)(\Gamma) \vdash^{\Sigma'} \mathbf{Sen}(\sigma)(\phi)$.

Lógica

[Mes89]

A *logic* is a structure of the form $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ satisfying the following conditions:

- $\mathsf{Sign}, \mathbf{Sen}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}$ is an entailment system,
- $\mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}$ is an institution, and
- the following *soundness* condition is satisfied: for any $\Sigma \in |\mathsf{Sign}|$, $\phi \in \mathbf{Sen}(\Sigma)$, $\Gamma \subseteq \mathbf{Sen}(\Sigma)$

$$\Gamma \vdash^{\Sigma} \phi \Longrightarrow \Gamma \models^{\Sigma} \phi \ .$$

A logic is *complete* if in addition the following condition is also satisfied: for any $\Sigma \in |\mathsf{Sign}|$, $\phi \in \mathbf{Sen}(\Sigma)$, $\Gamma \subseteq \mathbf{Sen}(\Sigma)$

$$\Gamma \vdash^{\Sigma} \phi \Longleftarrow \Gamma \models^{\Sigma} \phi \ .$$

Especificaciones estructuradas

[Bor02]

Dada una institución $\mathrm{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in \mathsf{Sign}} \rangle$ las especificaciones sobre Σ (denotada como Spec_Σ) se construyen de la siguiente forma:

- Dados $\Sigma \in |\mathsf{Sign}|$ y $\Gamma \subseteq \mathbf{Sen}(\Sigma)$, $\langle \Sigma, \Gamma \rangle$ es una especificación:

  $\mathbf{Sig}[\langle \Sigma, \Gamma \rangle] = \Sigma$,
  $\mathbf{Mod}[\langle \Sigma, \Gamma \rangle] = Mod_{\langle \Sigma, \Gamma \rangle}$,

- dadas dos SP_1 y SP_2 tal que $\mathbf{Sig}[SP_1] = \mathbf{Sig}[SP_2]$, $SP_1 \cup SP_2$ es una especificación tal que:

  $\mathbf{Sig}[SP_1 \cup SP_2] = \mathbf{Sig}[SP_1] = \mathbf{Sig}[SP_2]$,
  $\mathbf{Mod}[SP_1 \cup SP_2] = \mathbf{Mod}[SP_1] \cap \mathbf{Mod}[SP_2]$,

- dada SP una especificación tal que $\mathbf{Sig}[SP] = \Sigma$ y $\sigma : \Sigma \to \Sigma'$ un morfismo, **translate** SP **by** σ es una especificación tal que:

  $\mathbf{Sig}[\textbf{translate } SP \textbf{ by } \sigma] = \Sigma'$,
  $\mathbf{Mod}[\textbf{translate } SP \textbf{ by } \sigma] = \{\mathcal{M}' \in |\mathbf{Mod}(\Sigma')| \mid \mathbf{Mod}(\sigma)(\mathcal{M}') \in \mathbf{Mod}[SP]\}$,

- dada SP' una especificación tal que $\mathbf{Sig}[SP'] = \Sigma'$ y $\sigma : \Sigma \to \Sigma'$ un morfismo, **derive** SP' **by** σ es una especificación tal que:

  $\mathbf{Sig}[\textbf{derive } SP' \textbf{ by } \sigma] = \Sigma$,
  $\mathbf{Mod}[\textbf{derive } SP' \textbf{ by } \sigma] = \{\mathbf{Mod}(\sigma)(\mathcal{M}') \mid \mathcal{M}' \in |\mathbf{Mod}[SP']\}$.

Especificaciones estructuradas

[Bor02]

Consecuencia semántica

Dada una institución $\mathsf{I} = \langle \mathrm{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in \mathrm{Sign}} \rangle$ se define $\models^{\mathsf{I}}_\Sigma \subseteq \mathrm{Spec}_\Sigma \times \mathbf{Sen}(\Sigma)$ de la siguiente forma:

$$SP \models^{\mathsf{I}}_\Sigma \phi \text{ sii } \mathbf{Mod}[SP] \models^{\mathsf{I}}_\Sigma \phi$$

Especificaciones estructuradas

[Bor02]

Consecuencia sintáctica

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ se define $\vdash^{\mathsf{I}}_{\Sigma} \subseteq \text{Spec}_{\Sigma} \times \mathbf{Sen}(\Sigma)$ de la siguiente forma:

$$\frac{\{SP \vdash_{\Sigma} \phi_i\}_{i \in \mathcal{I}} \quad \{\phi_i\}_{i \in \mathcal{I}} \vdash^{\mathsf{I}}_{\Sigma} \phi}{SP \vdash_{\Sigma} \phi} \text{ (CR)}$$

$$\frac{\phi \in \Gamma}{\langle \Sigma, \Gamma \rangle \vdash_{\Sigma} \phi} \text{ (basic)}$$

$$\frac{SP_1 \vdash_{\Sigma} \phi}{SP_1 \cup SP_2 \vdash_{\Sigma} \phi} \text{ (sum\_1)}$$

$$\frac{SP_2 \vdash_{\Sigma} \phi}{SP_1 \cup SP_2 \vdash_{\Sigma} \phi} \text{ (sum\_2)}$$

$$\frac{SP \vdash_{\Sigma} \phi}{\textbf{translate } SP \textbf{ by } \sigma \vdash_{\Sigma'} \mathbf{Sen}(\sigma)(\phi)} \text{ (trans)}$$

$$\frac{SP' \vdash_{\Sigma'} \mathbf{Sen}(\sigma)(\phi)}{\textbf{derive } SP' \textbf{ by } \sigma \vdash_{\Sigma} \phi} \text{ (derive)}$$

con $\sigma : \Sigma \to \Sigma'$.

Sobre las relaciones de consecuencia

[Bor02]

Dada una lógica $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ y las familias de relaciones $\models^{\mathsf{I}}_{\Sigma} \subseteq \mathsf{Spec}_{\Sigma} \times \mathbf{Sen}(\Sigma)$ y $\vdash^{\mathsf{I}}_{\Sigma} \subseteq \mathsf{Spec}_{\Sigma} \times \mathbf{Sen}(\Sigma)$,

$\vdash^{\mathsf{I}}_{\Sigma}$ es *sound* con respecto a $\models^{\mathsf{I}}_{\Sigma}$ si y sólo si $\vdash^{\mathsf{I}}_{\Sigma} \subseteq \models^{\mathsf{I}}_{\Sigma}$.

$\vdash^{\mathsf{I}}_{\Sigma}$ es *complete* con respecto a $\models^{\mathsf{I}}_{\Sigma}$ si y sólo si $\models^{\mathsf{I}}_{\Sigma} \subseteq \vdash^{\mathsf{I}}_{\Sigma}$.

Forma normal

[Bor02]

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ y una especificación SP sobre Σ se dice que SP está en forma normal si es de la forma:

$$\mathbf{derive}\ \langle \Sigma, \Gamma \rangle\ \mathbf{by}\ \sigma$$

con $\sigma : \mathbf{Sig}[SP] \to \Sigma$.

Forma normal

[Bor02]

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ se define la función $\mathbf{nf} : \text{Spec}_{\Sigma} \longrightarrow \text{Spec}_{\Sigma}$ de la siguiente forma:

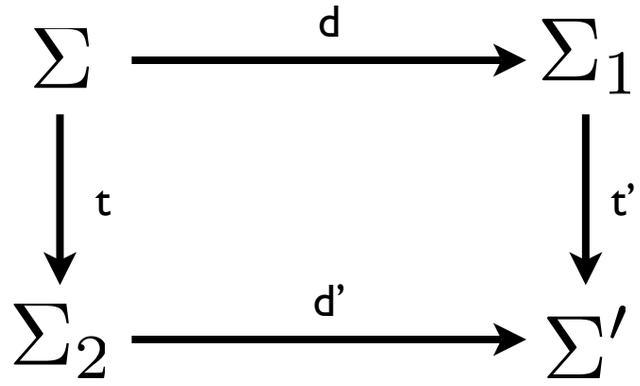
$$
\begin{array}{rcl}
\mathbf{nf}(\langle \Sigma, \Gamma \rangle) & = & \mathbf{derive}\ \langle \Sigma, \Gamma \rangle\ \mathbf{by}\ id_{\Sigma} \\
\mathbf{nf}(SP_1 \cup SP_2) & = & \mathbf{derive}\ \langle \Sigma, \mathbf{Sen}(\sigma_1')(\Gamma_1) \cup \mathbf{Sen}(\sigma_2')(\Gamma_2) \rangle\ \mathbf{by}\ \widehat{\sigma}\ \text{tal que}: \\
 & & \quad \mathbf{nf}(SP_i) = \mathbf{derive}\ \langle \Sigma, \Gamma_i \rangle\ \mathbf{by}\ \delta_i, \\
 & & \quad \widehat{\sigma} = \delta_1 \circ \sigma_2' = \delta_2 \circ \sigma_1',\ \text{y} \\
 & & \quad \sigma_1'\ \text{y}\ \sigma_2'\ \text{se obtienen del pushout.} \\
\mathbf{nf}(\mathbf{translate}\ SP\ \mathbf{by}\ \sigma) & = & \mathbf{derive}\ \langle \Sigma', \mathbf{Sen}(\sigma')(\Gamma) \rangle\ \mathbf{by}\ \widehat{\sigma}\ \text{tal que}: \\
 & & \quad \mathbf{nf}(SP) = \mathbf{derive}\ \langle \Sigma, \Gamma \rangle\ \mathbf{by}\ \delta,\ \text{y} \\
 & & \quad \sigma'\ \text{y}\ \widehat{\sigma}\ \text{se obtienen del pushout.} \\
\mathbf{nf}(\mathbf{derive}\ SP'\ \mathbf{by}\ \sigma) & = & \mathbf{derive}\ \langle \Sigma', \Gamma \rangle\ \mathbf{by}\ \sigma \circ \delta\ \text{tal que}: \\
 & & \quad \mathbf{nf}(SP) = \mathbf{derive}\ \langle \Sigma', \Gamma \rangle\ \mathbf{by}\ \delta
\end{array}
$$

Forma normal

[Bor02]

Weak amalgamation property

Una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ satisface la *weak amalgamation property* si para todo *pushout* en Sign de la forma



y para cada $\mathcal{M}_1 \in |\mathbf{Mod}(\Sigma_1)|$ y $\mathcal{M}_2 \in |\mathbf{Mod}(\Sigma_2)|$ tal que $\mathbf{Sen}(d)(\mathcal{M}_1) = \mathbf{Sen}(t)(\mathcal{M}_2)$ entonces existe $\mathcal{M}' \in |\mathbf{Mod}(\Sigma')|$ tal que $\mathcal{M}_1 = \mathbf{Sen}(t')(\mathcal{M}')$ y $\mathcal{M}_2 = \mathbf{Sen}(d')(\mathcal{M}')$.

Forma normal

Weak amalgamation property

(en resumen)

Una lógica $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ satisface la *weak amalgamation property* si todo *pushout* en Sign induce un *pullback* en Cat entre las categorías de modelos de las signaturas intervinientes en el *pushout*.

Forma normal

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ **si satisface la *weak amalgamation property*, entonces:**

$$\mathbf{nf}(SP) \equiv SP$$

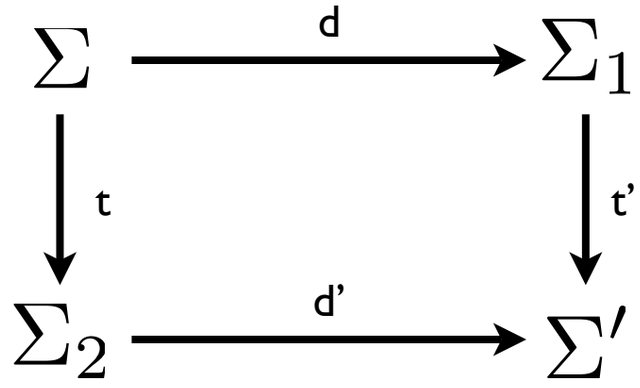
Dadas SP y SP' dos especificaciones, $SP \equiv SP'$ si y sólo si: $\mathbf{Sig}[SP] = \mathbf{Sig}[SP']$ y $\mathbf{Mod}[SP] = \mathbf{Mod}[SP']$

Forma normal

[Bor02]

Interpolation property

Una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ satisface la *interpolation property* si para todo *pushout* en Sign de la forma



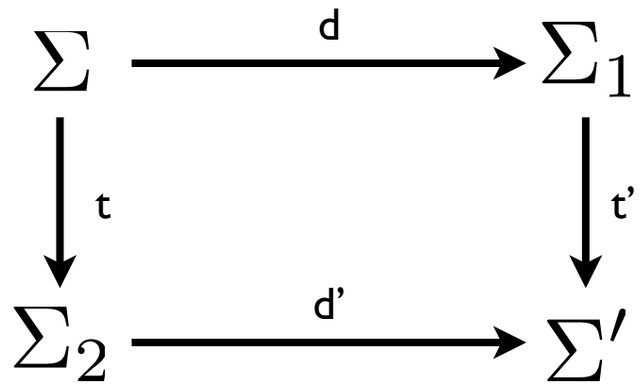
si $\mathbf{Sen}(t')(\phi_1) \models^{\Sigma'} \mathbf{Sen}(d')(\phi_2)$, entonces existe $\phi \in \mathbf{Sen}(\Sigma)$ tal que $\phi_1 \models^{\Sigma_1} \mathbf{Sen}(d)(\phi)$ y $\mathbf{Sen}(t)(\phi) \models^{\Sigma_2} \phi_2$.

Forma normal

[Bor02]

Weak interpolation property

Una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ satisface la *interpolation property* si para todo *pushout* en Sign de la forma



si $\mathbf{Sen}(t')(\phi_1) \models^{\Sigma'} \mathbf{Sen}(d')(\phi_2)$, entonces existe $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ tal que $\phi_1 \models^{\Sigma_1} \mathbf{Sen}(d)(\Gamma)$ y $\mathbf{Sen}(t)(\Gamma) \models^{\Sigma_2} \phi_2$.

Sobre las relaciones de consecuencia

[Bor02]

Soundness

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$
$\vdash_{\Sigma}$ es *sound* con respecto a $\models_{\Sigma}$.

Sobre las relaciones de consecuencia

[Bor02]

Completeness

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ **si tiene conjunción infinita e implicación y satisface:**

- la *interpolation property*,
- la *weak amalgamation property*, y
- la relación $\vdash^{\Sigma}$ es completa,

entonces $\vdash_{\Sigma}$ es completa con respecto a $\models_{\Sigma}$.

Sobre las relaciones de consecuencia

[Bor02]

Completeness

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ **si** tiene conjunción e implicación y satisface:

- la *weak interpolation property*,
- la *weak amalgamation property*,
- la propiedad de compacidad, y
- la relación $\vdash^{\Sigma}$ es completa,

entonces $\vdash_{\Sigma}$ es completa con respecto a $\models_{\Sigma}$.

Sobre las relaciones de consecuencia

[Bor02]

Completeness

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ **si tiene conjunción infinita e implicación y satisface:**

- la *weak interpolation property*,
- la *weak amalgamation property*, y
- la relación $\vdash^{\Sigma}$ es completa,

entonces $\vdash_{\Sigma}$ es completa con respecto a $\models_{\Sigma}$.

Sobre las relaciones de consecuencia

[Bor02]

Completeness

Dada una lógica $\langle \text{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\text{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\text{Sign}|} \rangle$ **si** tiene conjunción e implicación y satisface:

- la *interpolation property*,
- la *weak amalgamation property*,
- las especificaciones son finitas, y
- la relación $\vdash^{\Sigma}$ es completa,

entonces $\vdash_{\Sigma}$ es completa con respecto a $\models_{\Sigma}$.

Resumiendo

Hemos aprendido cómo estructurar especificaciones de forma de dar soporte teórico a una parte importante de la metodología con la que estas son contruídas.

Además presentamos condiciones a través de las cuales se puede obtener un cálculo completo para especificaciones estructuradas.

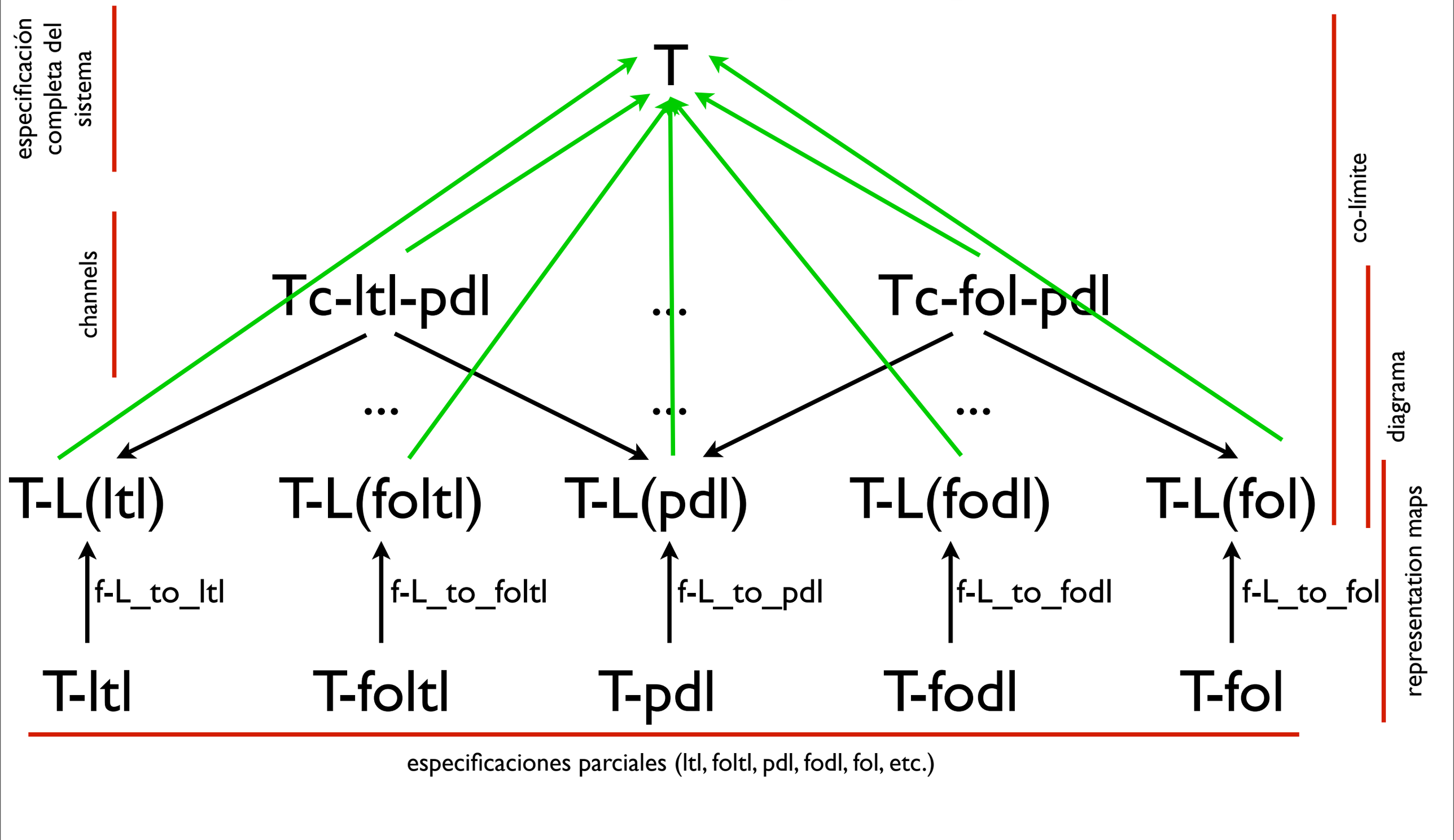
E identificamos ciertos límites impuestos por estas condiciones que hacen que dicho cálculo sólo sirva para una cantidad limitada de instituciones... pero no avanzaremos sobre la solución de esto. **:-)**

Representación de especificaciones

[Bor02]

Ya discutimos en profundidad las ventajas de tener especificaciones estructuradas, pero ¿cómo se lleva esto con nuestra mirada sobre el problema de trabajar con especificaciones heterogéneas?

(recordando) La propuesta...



Representación de especificaciones

[Bor02]

Dadas dos lógicas $\mathsf{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models_\Sigma\}_{\Sigma \in \mathsf{Sign}} \rangle$ e $\mathsf{I}' = \langle \mathsf{Sign}', \mathbf{Sen}', \mathbf{Mod}', \{\models'_\Sigma\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ y un *representation map* $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod} \rangle : \mathsf{I} \to \mathsf{I}'$ se define la familia de funciones $\{\widehat{\gamma}_\Sigma : \mathsf{Spec}_\Sigma \to \mathsf{Spec}_{\gamma^{Sign}(\Sigma)}\}_{\Sigma \in |\mathsf{Sign}|}$ de la siguiente forma:

- si $SP = \langle \Sigma, \Gamma \rangle$, entonces $\widehat{\gamma}_\Sigma(SP) = \langle \gamma^{Sign}(\Sigma), \gamma^{Sen}_\Sigma(\Gamma) \rangle$,
- si $SP = SP_1 \cup SP_2$, entonces $\widehat{\gamma}_\Sigma(SP) = \widehat{\gamma}_\Sigma(SP_1) \cup \widehat{\gamma}_\Sigma(SP_2)$,
- si $SP = \mathbf{translate}\ SP'\ \mathbf{by}\ \sigma : \Sigma' \to \Sigma''$, entonces
  $\widehat{\gamma}_\Sigma(SP) = \mathbf{translate}\ \widehat{\gamma}_{\Sigma'}(SP')\ \mathbf{by}\ \gamma^{Sign}(\sigma : \Sigma' \to \Sigma'')$,
- si $SP = \mathbf{derive}\ SP'\ \mathbf{by}\ \sigma : \Sigma \to \Sigma'$, entonces
  $\widehat{\gamma}_\Sigma(SP) = \mathbf{derive}\ \widehat{\gamma}_{\Sigma'}(SP')\ \mathbf{by}\ \gamma^{Sign}(\sigma : \Sigma \to \Sigma')$.

Representación de especificaciones

[Bor02]

Preservación de la forma normal

Dado un *representation map* $\gamma : \mathsf{I} \to \mathsf{I}'$ y una Σ-especificación SP sobre I, si $\gamma^{Sign} : \mathsf{Sign} \to \mathsf{Sign}'$ preserva *pushouts*, entonces $\mathbf{nf}(\widehat{\gamma}_\Sigma(SP)) = \widehat{\gamma}_\Sigma(\mathbf{nf}(SP))$.

Representación de especificaciones

[Bor02]

Weak amalgamation property

Un representation map $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod} \rangle : \mathrm{I} \to \mathrm{I}'$ satisface la *weak amalgamation property* si y sólo si para todo $\sigma : \Sigma \to \Sigma'$ en Sign, $\mathcal{M} \in |\mathbf{Mod}(\Sigma)|$ y $\mathcal{M}' \in |\mathbf{Mod}'(\gamma^{Sign}(\Sigma'))|$, si $\gamma^{Mod}_{\Sigma'}(\mathcal{M}') = \mathbf{Mod}(\sigma)(\mathcal{M})$, entonces existe $\mathcal{M}'' \in |\mathbf{Mod}'(\gamma^{Sign}(\Sigma))|$ tal que $\gamma^{Mod}_{\Sigma'}(\mathcal{M}'') = \mathcal{M}$.

Representación de especificaciones

Weak amalgamation property

(en resumen)

Un representation map $\langle \gamma^{Sign}, \gamma^{Sen}, \gamma^{Mod} \rangle : \mathrm{I} \longrightarrow \mathrm{I}'$ satisface la *weak amalgamation property* si y sólo si garantiza que existen los *pullbacks* entre los modelos que forman parte de las clases de modelos de dos signaturas relacionadas y sus representaciones.

Representación de especificaciones

[Bor02]

Preservación de la forma normal

Dado un *representation map* $\gamma : \mathsf{I} \to \mathsf{I}'$ y una Σ-especificación SP sobre I, si $\gamma^{Sign} : \mathsf{Sign} \to \mathsf{Sign}'$ preserva *pushouts* y γ satisface la *weak amalgamation property*, entonces $\widehat{\gamma}_\Sigma(SP) \equiv \widehat{\gamma}_\Sigma(\mathbf{nf}(SP))$.

Representación de especificaciones

A partir de propiedades como la *weak amalgamation property* y otras que no veremos aquí, se puede estudiar la relación que existe entre las clases de modelos de una especificación dada y su representación.

Es decir, si SP es un especificación sobre I y $\gamma : \mathrm{I} \to \mathrm{I}'$ es un *representation map*, entonces nos gustaría comparar $\mathbf{Mod}[SP]$ con $\mathbf{Mod}[\widehat{\gamma}_{\Sigma}(SP)]$.

Representación de especificaciones

A partir de propiedades como la *weak amalgamation property* y otras que no veremos aquí, se puede estudiar la relación que existe entre las clases de modelos de una especificación dada y su representación.

Es decir, si SP es un especificación sobre I y $\gamma : \mathrm{I} \to \mathrm{I}'$ es un *representation map*, entonces nos gustaría comparar $\mathbf{Mod}[SP]$ con $\mathbf{Mod}[\widehat{\gamma}_{\Sigma}(SP)]$.

¡Nos encantaría, pero no vamos a hacerlo!

Representación de especificaciones

Un detalle pendiente

De la misma forma en la que extendimos los *representation maps* entre instituciones a *maps* entre instituciones para tomar cuenta de las teorías, en lugar de sólo observar las signaturas, puede extenderse la definición de **representación de especificaciones** vista anteriormente a **maps entre especificaciones** a fin de tomar cuenta de los axiomas particulares que la institución más poderosa requiera agregar a cada teoría.

Resumen
(lo que hemos aprendido)

- Hemos formalizado el concepto de lógica, a través del concepto de institución poniendo de relieve qué es lo que significa satisfactibilidad independientemente de los símbolos usados,
- Hemos formalizado la relación entre lógicas a través del concepto de *representation map* independizando el concepto de satisfactibilidad de la estructura sintáctica de una lógica particular,
- Hemos integrado estas herramientas en un ambiente que permite manipular especificaciones heterogéneas,

Resumen
(lo que hemos aprendido)

- Hemos formalizado el concepto de especificación estructurada para dar soporte a la metodología con la que en general se desarrollan las descripciones formales,
- Hemos presentado condiciones para la existencia de un cálculo completo para especificaciones estructuradas,
- Y finalmente, hemos extendido el concepto de representabilidad para especificaciones estructuradas.

Bibliografía

[End72] Herbert B. Enderton, *A mathematical introduction to logic*, Academic Press, 1972. **P**

[McL71] Saunder McLane, *Categories for working mathematicians*, Graduate Texts in Mathematics, Springer-Verlag, Berlin, Germany, 1971. **P**

[Fia05] José Luis Fiadeiro, *Categories for software engineering*, Springer-Verlag, 2005. **P**

[BS81] Stanley Burris and H. P. Sankappanavar, *A course in universal algebra*, Graduate Texts in Mathematics, Springer-Verlag, Berlin, Germany, 1981. **D**

[GB84] Joseph A. Goguen and Rod M. Burstall, *Introducing institutions*, Proceedings of the Carnegie Mellon Workshop on Logic of Programs (Edmund M. Clarke and Dexter Kozen, eds.), Lecture Notes in Computer Science, vol. 184, Springer-Verlag, 1984, pp. 221–256. **P**

[GB92] Joseph A. Goguen and Rod M. Burstall, *Institutions: abstract model theory for specification and programming*, Journal of the ACM 39 (1992), no. 1, 95–146. **D**

[Fia96] José Luis Fiadeiro, *On the emergence of properties in component-based systems*, Proceedings of the 1996 Algebraic Methodology and Software Technology – AMAST 96 (Munich, Germany) (Martin Wirsing and Maurice Nivat, eds.), Lecture Notes in Computer Science, vol. 1101, Springer-Verlag, July 1996. **D**

[Mes89] José Meseguer, *General logics*, Proceedings of the Logic Colloquium ’87 (Granada, Spain) (Heinz-Dieter Ebbinghaus, José Fernandez-Prida, Manuel Garrido, Daniel Lascar, and Mario Rodríguez Artalejo, eds.), vol. 129, North Holland, 1989, pp. 275–329. **D**

[Tar96] Andrzej Tarlecki, *Moving between logical systems*, Selected papers from the 11th Workshop on Specification of Abstract Data Types Joint with the 8th COMPASS Workshop on Recent Trends in Data Type Specification (Magne Haveraaen, Olaf Owe, and Ole-Johan Dahl, eds.), Lecture Notes in Computer Science, vol. 1130, Springer-Verlag, 1996, pp. 478–502. **D**

[FBH97] Marcelo F. Frias, Gabriel A. Baum, and Armando M. Haeberer, *Fork algebras in algebra, logic and computer science*, Fundamenta Informaticae 32 (1997), 1–25. **P**

[Gyu97] Viktor Gyuris, *A short proof of representability of fork algebra*, Theoretical Computer Science 188 (1997), no. 1–2, 211–220. **D**

Bibliografía

[FHV97] Marcelo F. Frias, Armando Haeberer, Paulo A.S. Veloso, *A finite axiomatization for fork algebras*, Logic Journal of the IGPL 5 (1997), no. 3, 311–319. **D**

[Fri02] Marcelo F. Frias, *Fork algebras in algebra, logic and computer science*, Advances in logic, vol. 2, World Scientific Publishing Co., Singapore, 2002. **D**

[FO98] Marcelo F. Frias and Ewa Orlowska, *Equational reasoning in non-classical logics*, Journal of Applied Non-classical Logics 8 (1998), no. 1–2, 27–66. **D**

[FBM02] Marcelo F. Frias, Gabriel A. Baum, and Tomas S. E. Maibaum, *Interpretability of first-order dynamic logic in a relational calculus*, Proceedings of the 6th. Conference on Relational Methods in Computer Science (RelMiCS) - TARSKI (Oisterwijk, The Netherlands) (Harrie de Swart, ed.), Lecture Notes in Computer Science, vol. 2561, Springer-Verlag, October 2002, pp. 66–80. **D**

[FL03] Marcelo F. Frias and Carlos G. Lopez Pombo, *Time is on my side*, in Rudolf Berghammer and Bernhard Möller (eds.), 7th. conference on relational methods in computer science (RelMiCS) - 2nd. international workshop on applications of kleene algebra, Malente, Germany, May 2003., pp. 105–111. **D**

[FL06] Marcelo F. Frias and Carlos G. Lopez Pombo, *Interpretability of first-order linear temporal logics in fork algebras*, Journal of Logic and Algebraic Programming 66 (2006), no. 2, 161–184. **D**

[FGLR05] Marcelo F. Frias, Juan P. Galeotti, Carlos G. Lopez Pombo, and Mario Roman, *Fork algebra as a formalism to reason across behavioral specifications (extended abstract)*, Proceedings of the 8th. Conference on Relational Methods in Computer Science (RelMiCS) - 3nd. International Workshop on Applications of Kleene Algebra (St. Catharines, Ontario, Canada) (Ivo Düntsch and Michael Winter, eds.), February 2005, pp. 61–68. **D**

[LF06] Carlos G. Lopez Pombo and Marcelo F. Frias, *Fork algebras as a suffi**ciently rich universal institution***, Proceedings of 11th International Conference on Algebraic Methodology and Software Technology, AMAST 2006 (Kuressaare, Estonia) (Michael Johnson and Varmo Vene, eds.), Lecture Notes in Computer Science, vol. 4019, Springer-Verlag, July, 5–8 2006, pp. 235–247. **D**

Bibliografía

[Lop07] Carlos G. Lopez Pombo. *Fork algebras as a tool for reasoning across heterogeneous specifications*. PhD thesis, Departamento de Computacion, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, 2007. Promotor: Marcelo F. Frias. **D**

[Bor02] Tomasz Borzyszkowski. *Logical systems for structured specifications*. Theoretical Computer Science, 286:197–245, 2002. **D**