Bachelor's thesis - final talk:

Organizing a Library of Higher Order Problems

by Julian Backes on April 6, 2009

Advisor: Chad Brown

Supervisor: Gert Smolka

Contents

- Recap from the first two talks
  - Our problem
  - Signature/Presentation/Provability
  - Morphisms
- The proof of the Presentation Lemma
- Imports
- Implementation
  - A datastructure for storing trees
  - Reducing memory and time consumption
- Demonstration
- Future Work

Recap

The story so far...

Our problem

- The context: Proofs in Jitpro
- Goal: Reusing existing "theories" and proven claims
- Problem: Combining different small theories to bigger, more powerful theories
- Example:

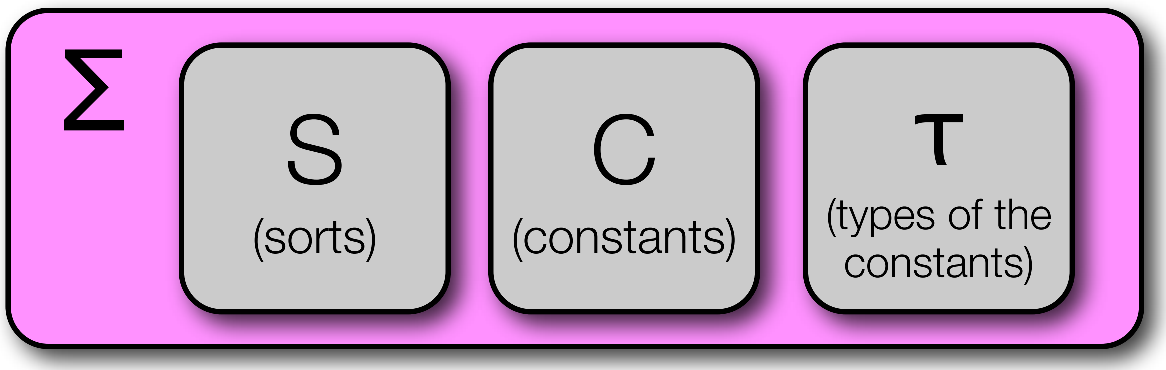
```
sort I; // set elements
var x: I;
var S, T: I B; // subsets
term union = \S T x.S x | T x; // definition of union

sort V; // vertices
var v1, v2, v3: V;
const E: V V B; // edges
axiom !v1 v2. (E v1 v2) -> (E v2 v1); // undirected graph

claim !v1, v2, v3. (E v1 v3) ->
                   (union (E v1) (E v2)) v3
```

Our problem

- The context: Proofs in Jitpro
- Goal: Reusing existing "theories" and proven claims
- Problem: Combining different small theories to bigger, more powerful theories
- Example:

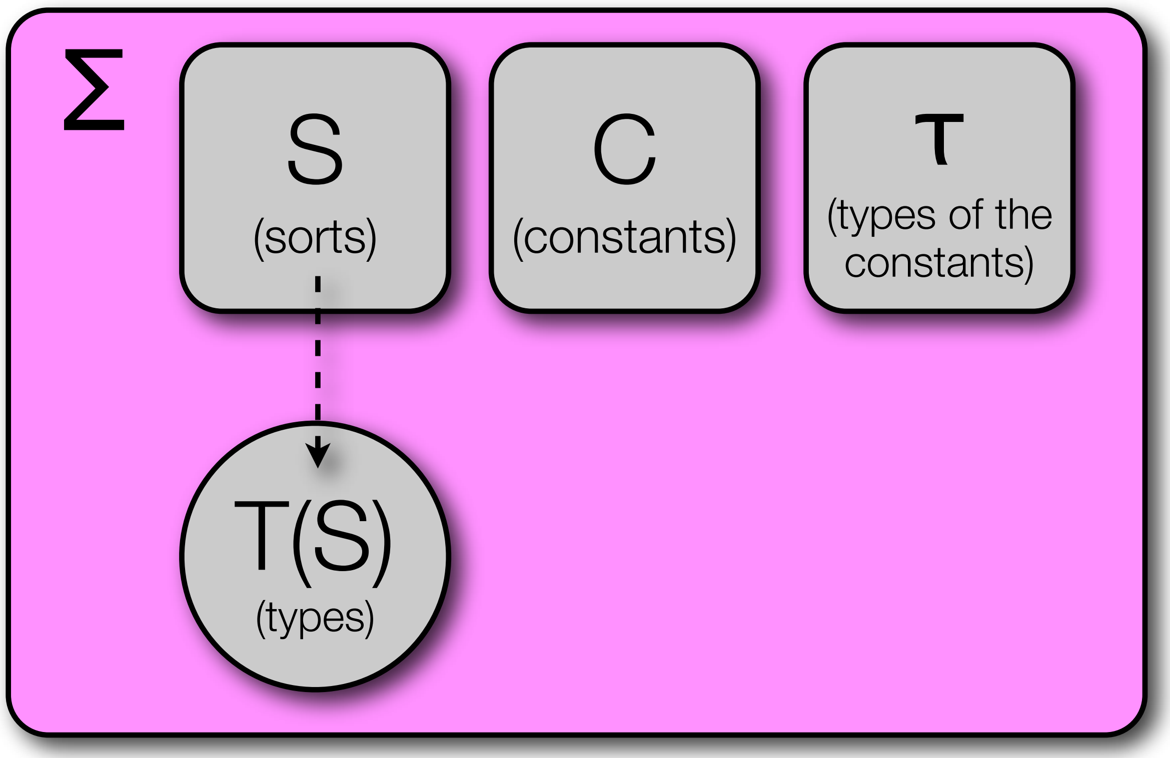
```
sort I; // set elements
var x: I;
var S, T: I B; // subsets
term union = \S T x.S x | T x; // definition of union

sort V; // vertices
var v1, v2, v3: V;
const E: V V B; // edges
axiom !v1 v2. (E v1 v2) -> (E v2 v1); // undirected graph

claim !v1, v2, v3. (E v1 v3) ->
                   (union (E v1) (E v2)) v3
```

Signatures / Terms



$\alpha \in$ Sorts

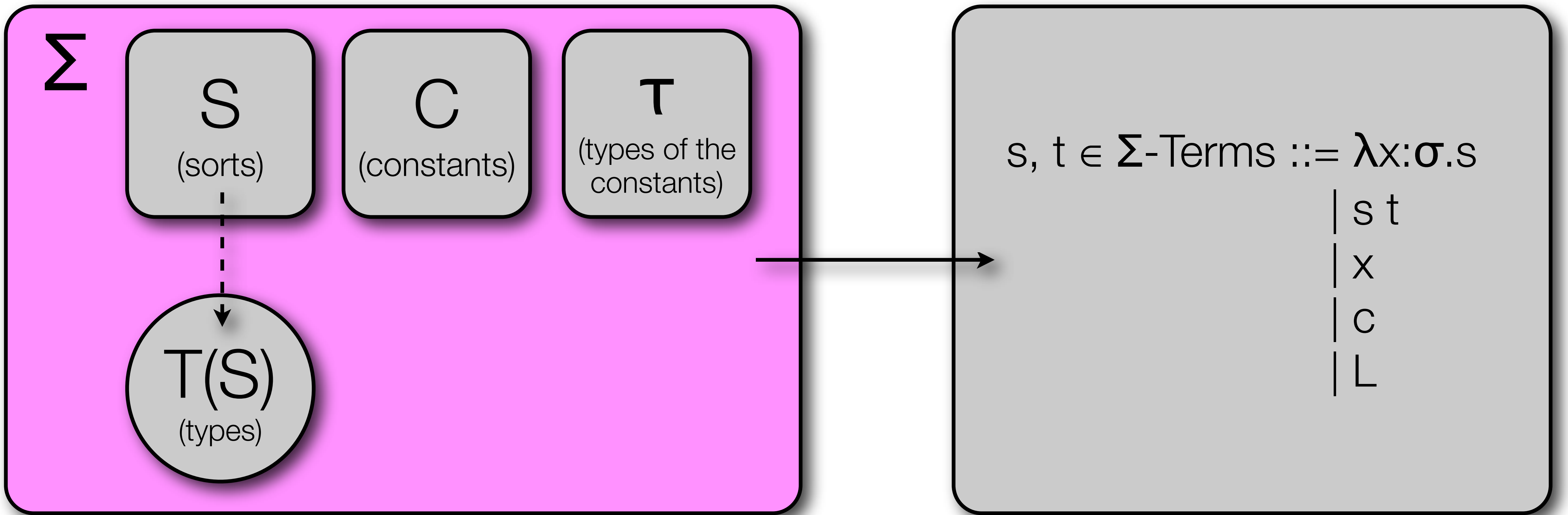
Signatures / Terms



$\alpha \in$ Sorts

$\sigma, \tau \in$ Types ::= $\alpha \mid \sigma\ \tau$

Signatures / Terms

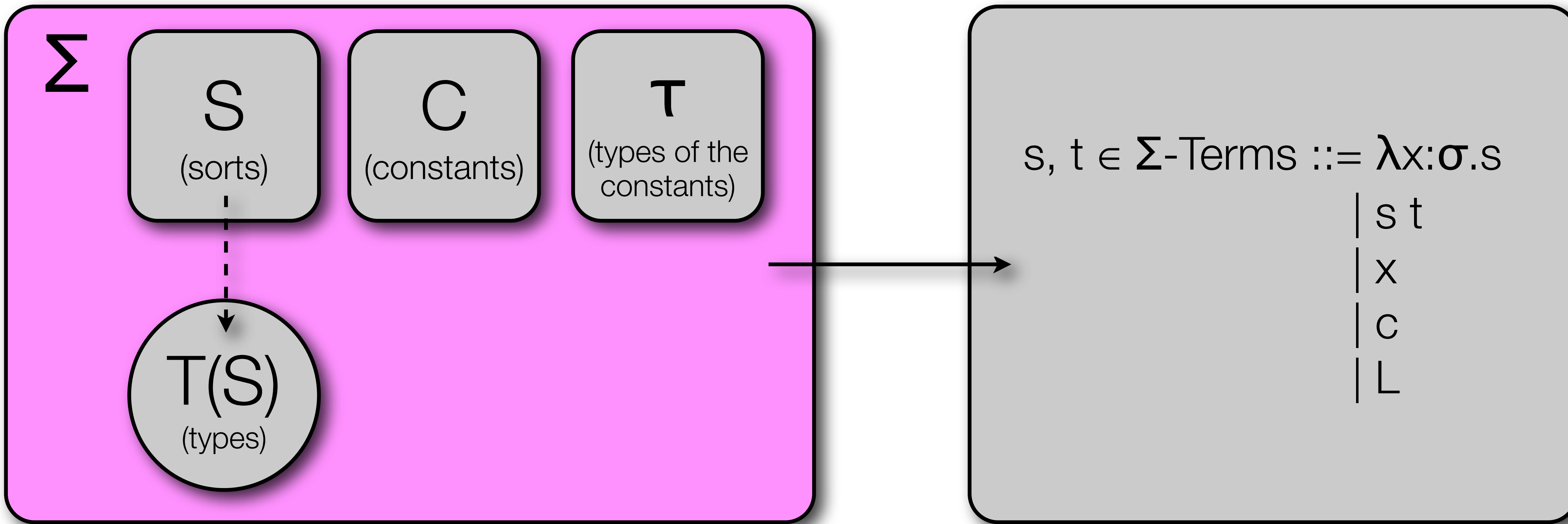


α ∈ Sorts

σ, τ ∈ Types ::= α | σ τ

C ∈ Contexts ::= [ ] | C s | s C | λx:σ.C

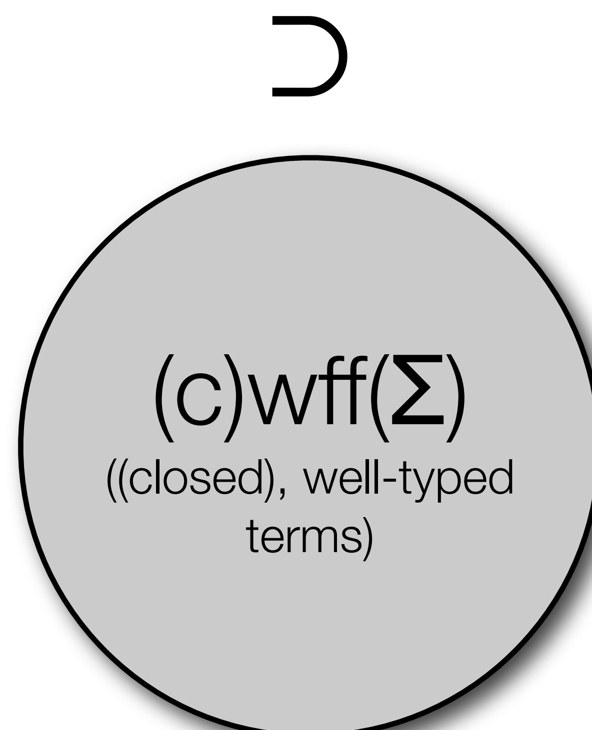
Signatures / Terms



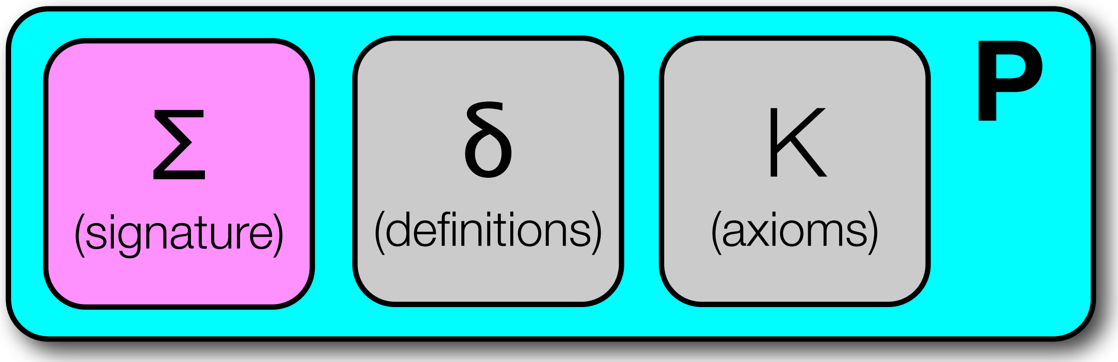
$\alpha \in$ Sorts

$\sigma, \tau \in$ Types ::= $\alpha \mid \sigma\ \tau$

$C \in$ Contexts ::= [ ] | C s | s C | $\lambda x{:}\sigma.C$



Presentations



Extended Proof System

- The proof system of Jitpro is defined by a set of basic refutation rules
- The rules depend on a signature, for example:

$$\textsc{Closed}\ \frac{}{A, \bot \vdash \bot}$$

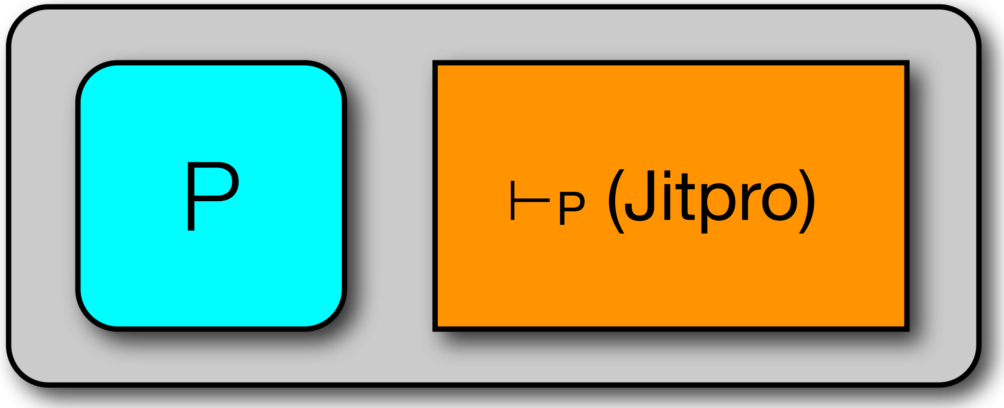
- Given a presentation P = {Σ, δ, K}, we extend the proof system by two additional presentation dependent rules:

$$\textsc{Axiom}_{\mathcal{P}}\ \frac{A, k \vdash \bot}{A \vdash \bot}\ \text{if } k \in \mathcal{K} \qquad \textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta)$$

- We call this proof system $\vdash_P$

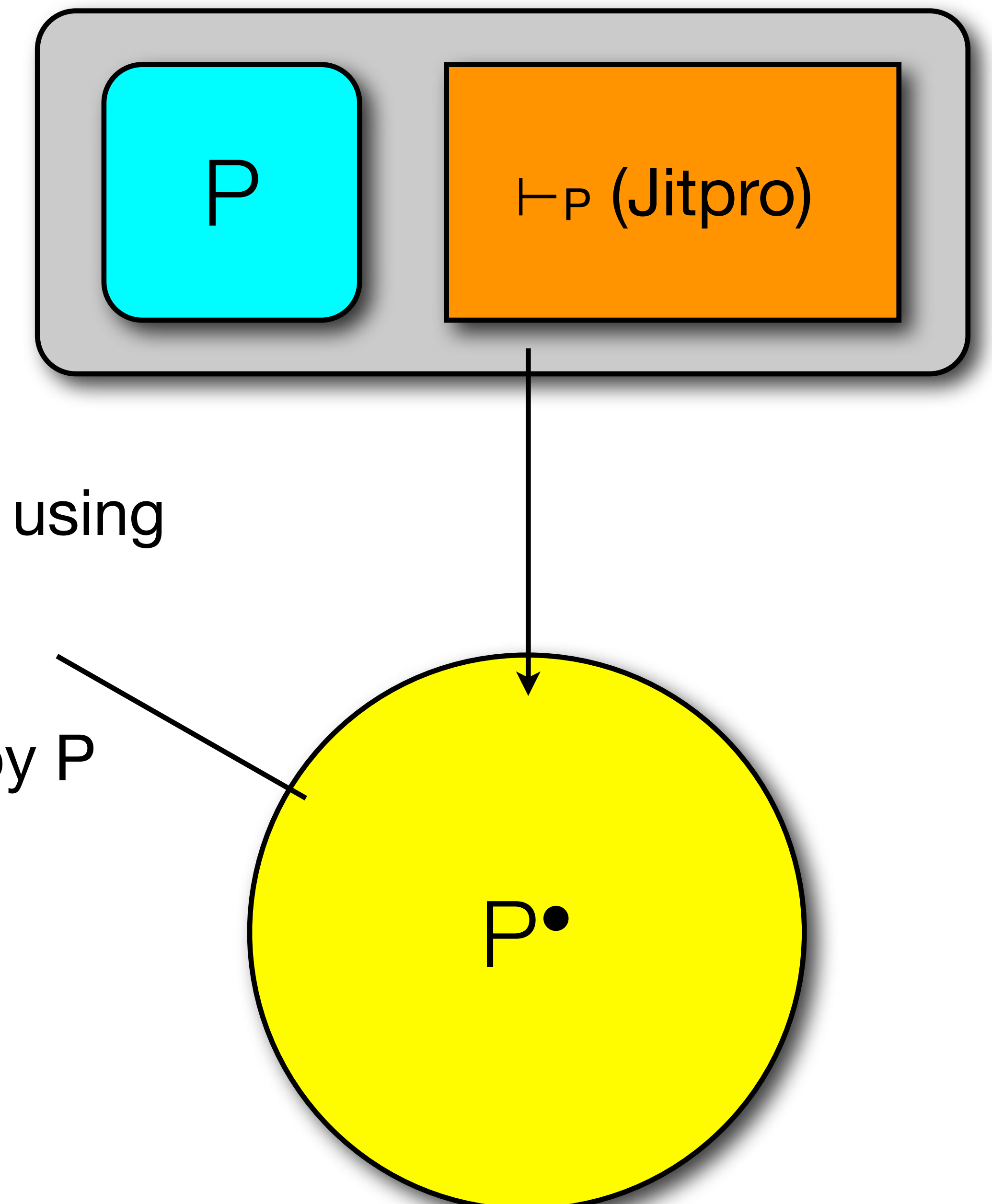
Closure / Theory

- Given a presentation P = {Σ, δ, K}, a claim $c \in cwff_B(\Sigma)$ is provable iff $\neg c \vdash \bot$ is provable using $\vdash_P$ (Jitpro)



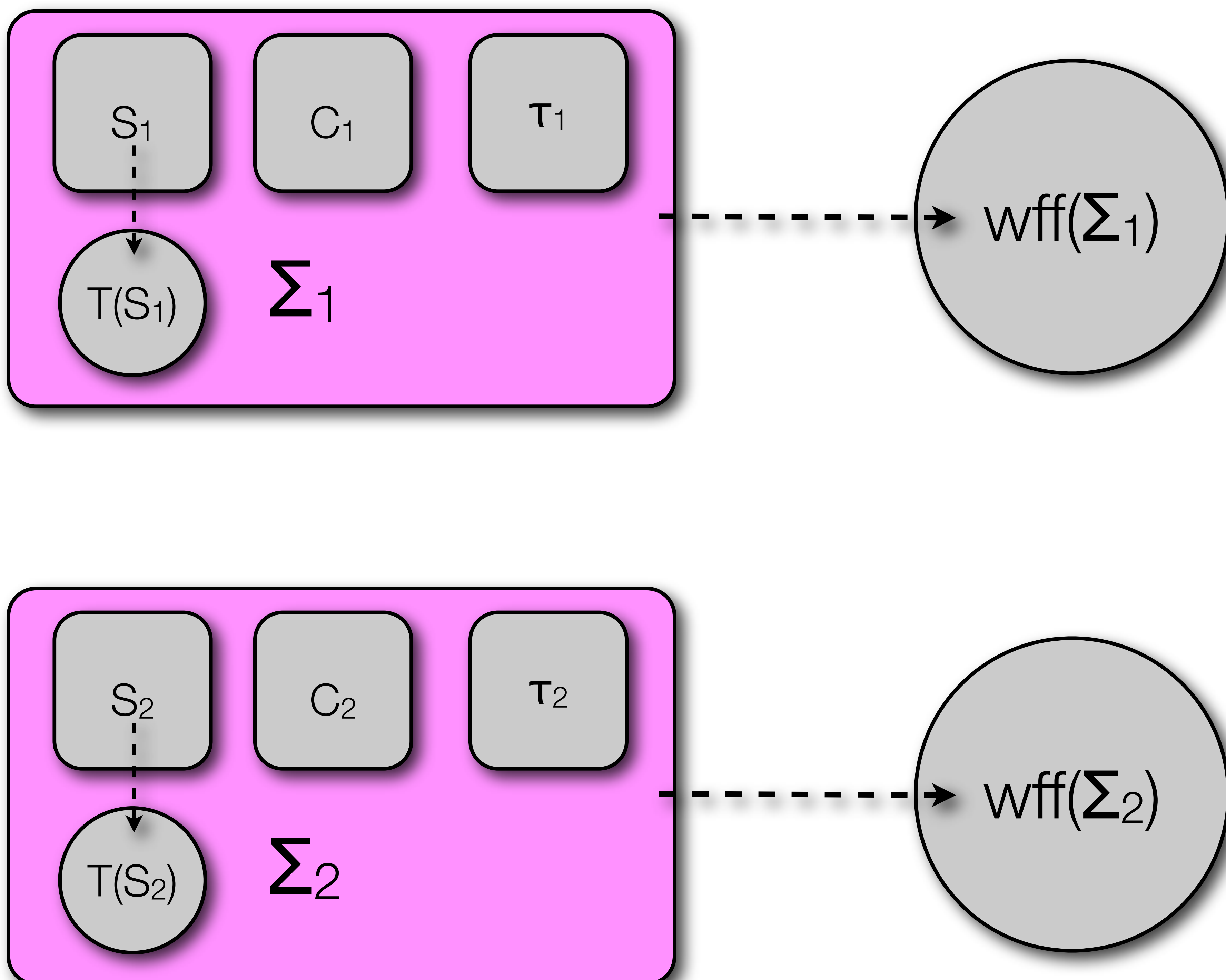
Closure / Theory

- Given a presentation $P = \{\Sigma, \delta, K\}$, a claim $c \in cwff_B(\Sigma)$ is provable iff $\neg c \vdash \bot$ is provable using $\vdash_P$ (Jitpro)

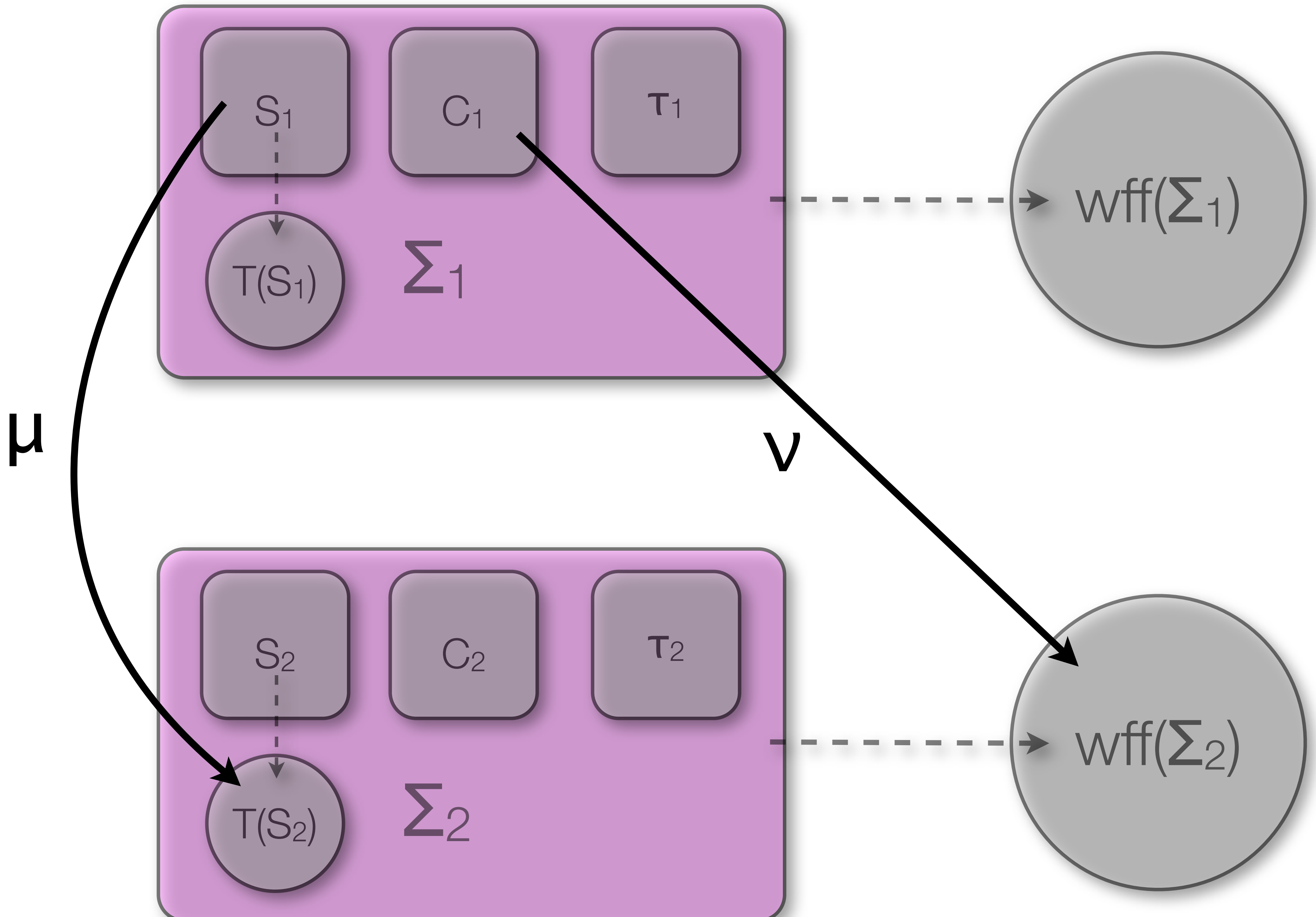


- all claims provable using $\vdash_P$
- closure of P
- theory presented by P

Signature morphisms (idea)



Signature morphisms (idea)



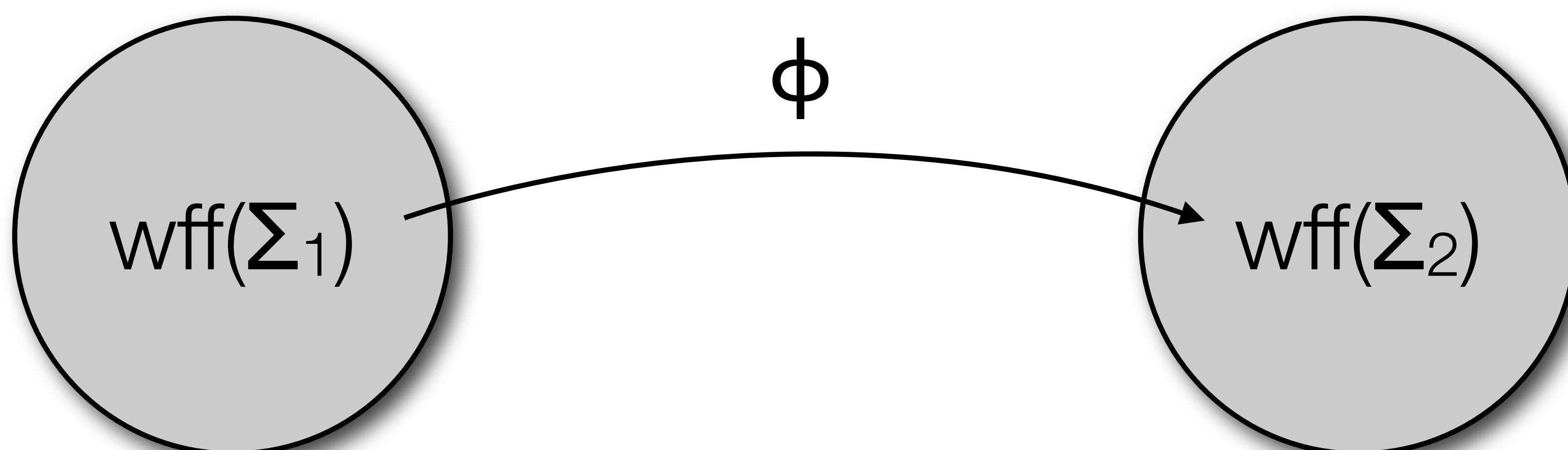
Signature morphisms ctd

- Let Σ_1, Σ_2 and $\phi = (\mu, \nu)$ be given
- Recursively define $\mu^\bullet$ on types using μ
- Recursively define $\nu^\bullet$ on terms using $\mu^\bullet$ and ν
- Recursively define $\nu^{\bullet\bullet}$ on contexts using $\nu^\bullet$

- ϕ is a ***signature morphism*** from Σ_1 and Σ_2

Signature morphisms ctd

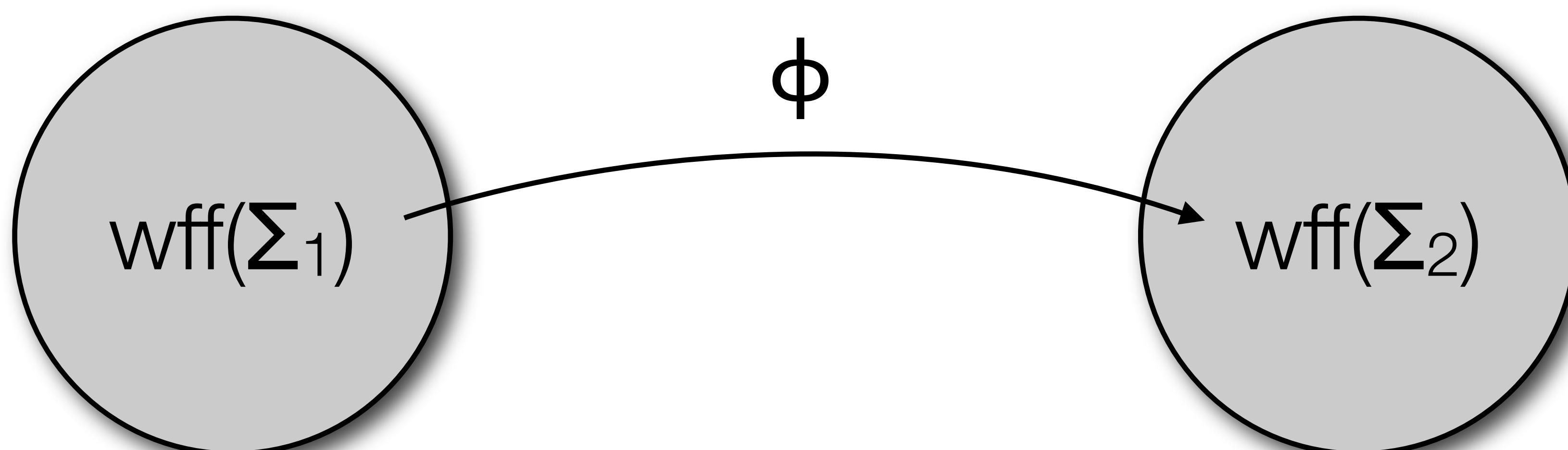
- Let Σ_1, Σ_2 and $\phi = (\mu, \nu)$ be given
- Recursively define $\mu^\bullet$ on types using μ
- Recursively define $\nu^\bullet$ on terms using $\mu^\bullet$ and ν
- Recursively define $\nu^{\bullet\bullet}$ on contexts using $\nu^\bullet$



- ϕ is a ***signature morphism*** from Σ_1 and Σ_2

Signature morphisms ctd

- Let Σ_1, Σ_2 and $\phi = (\mu, \nu)$ be given
- Recursively define $\mu^\bullet$ on types using μ
- Recursively define $\nu^\bullet$ on terms using $\mu^\bullet$ and ν
- Recursively define $\nu^{\bullet\bullet}$ on contexts using $\nu^\bullet$



- ϕ is a ***signature morphism*** from Σ_1 and Σ_2

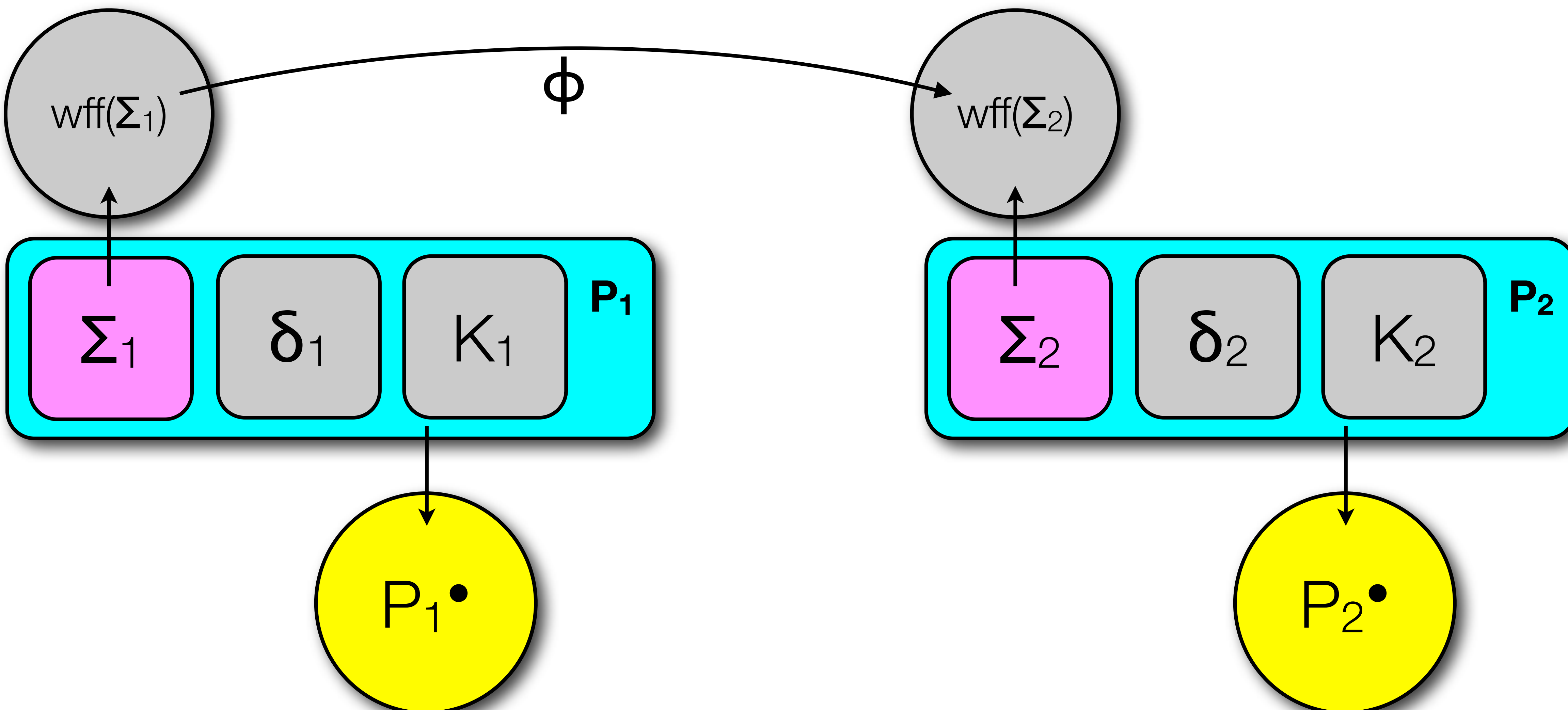
Theory morphisms

- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \to \Sigma_2$ be given

- ϕ is a ***theory morphism*** iff $\phi(P_1^\bullet) \subseteq P_2^\bullet$ (preservation of provability)

Theory morphisms

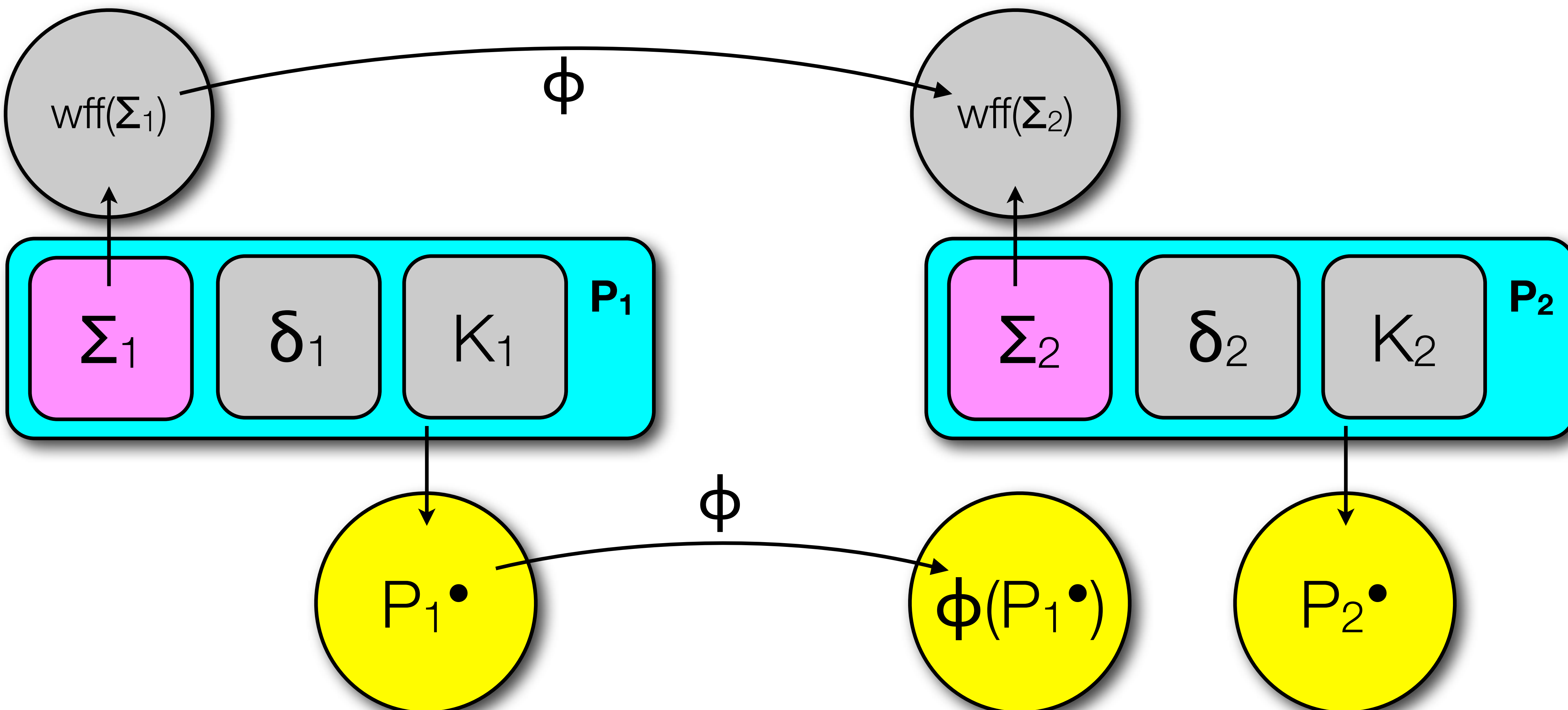
- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \rightarrow \Sigma_2$ be given



- ϕ is a ***theory morphism*** iff $\phi(P_1^\bullet) \subseteq P_2^\bullet$ (preservation of provability)

Theory morphisms

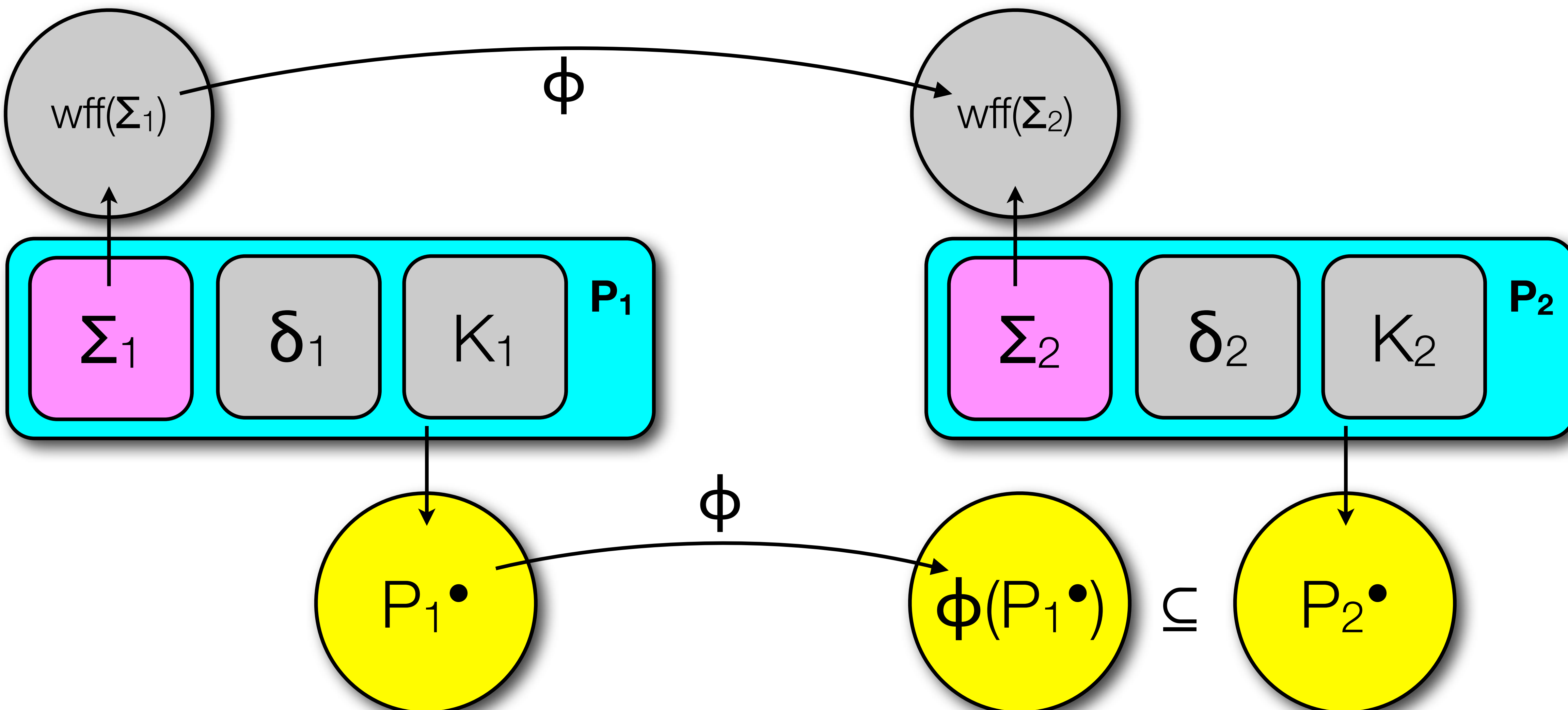
- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \to \Sigma_2$ be given



- ϕ is a ***theory morphism*** iff $\phi(P_1^\bullet) \subseteq P_2^\bullet$ (preservation of provability)

Theory morphisms

- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \to \Sigma_2$ be given



- ϕ is a ***theory morphism*** iff $\phi(P_1^\bullet) \subseteq P_2^\bullet$ (preservation of provability)

Theory morphisms ctd

- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \to \Sigma_2$ be given
- Problem: If we want to show that ϕ is a theory morphism, i.e. that we can reuse existing proofs, we first have to reprove everything which can be quite a lot of work.

Theory morphisms ctd

- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \rightarrow \Sigma_2$ be given

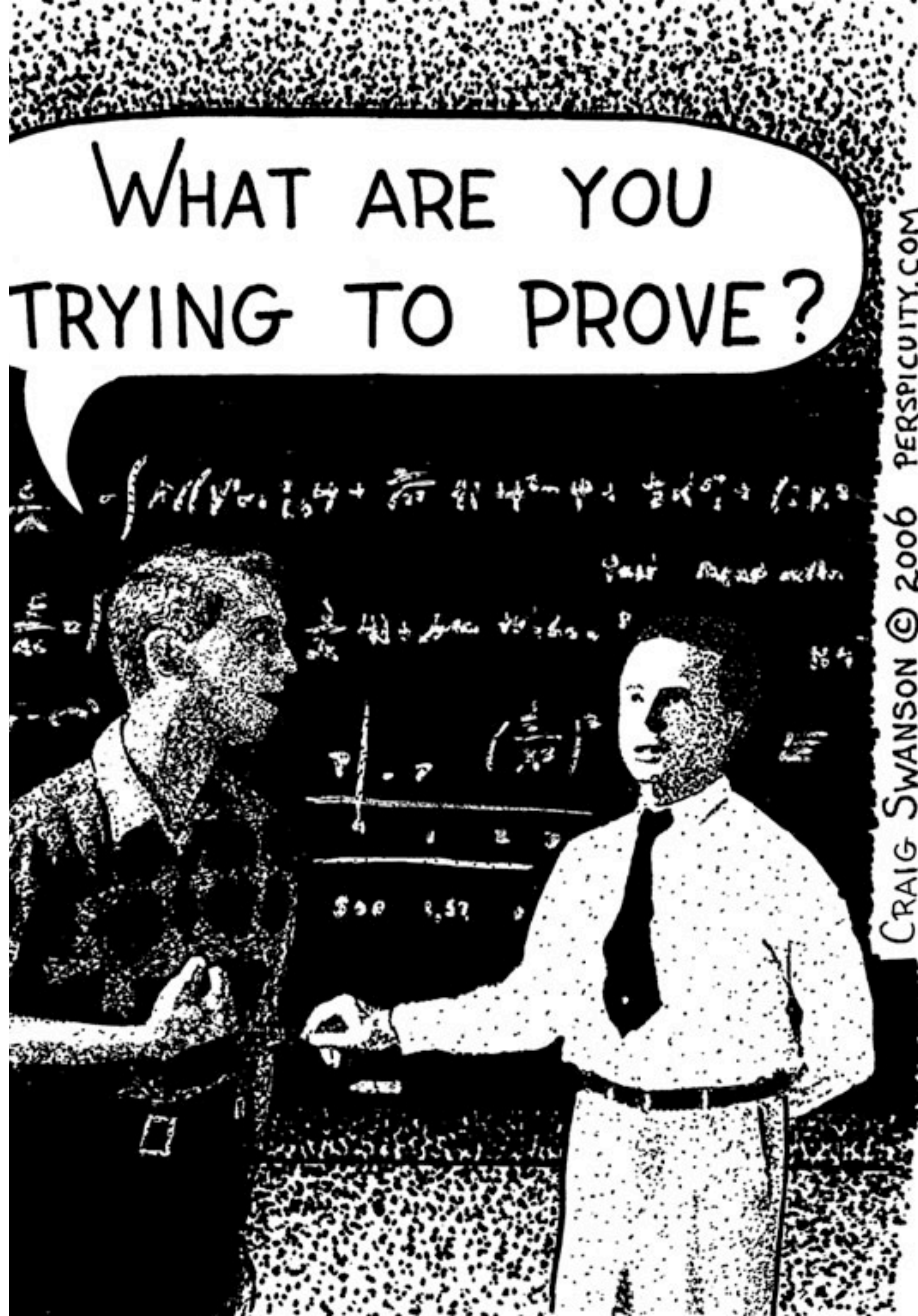
- Problem: If we want to show that ϕ is a theory morphism, i.e. that we can reuse existing proofs, we first have to reprove everything which can be quite a lot of work.

- Fortunately: **Presentation Lemma**: If $\phi(k) \in P_2^\bullet$ for all $k \in K_1$ and $(\phi(d) = \phi(\delta_1(d))) \in P_2^\bullet$ for all $d \in Dom(\delta)$ then ϕ is a theory morphism from $P_1^\bullet$ to $P_2^\bullet$.

Theory morphisms ctd

- Let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \rightarrow \Sigma_2$ be given
- Problem: If we want to show that ϕ is a theory morphism, i.e. that we can reuse existing proofs, we first have to reprove everything which can be quite a lot of work.
- Fortunately: **Presentation Lemma**: If $\phi(k) \in P_2^\bullet$ for all $k \in K_1$ and $(\phi(d) = \phi(\delta_1(d))) \in P_2^\bullet$ for all $d \in Dom(\delta)$ then ϕ is a theory morphism from $P_1^\bullet$ to $P_2^\bullet$.
- => It is enough to check all knowns and definitions (which <u>can</u> be trivial as we will later see)

The Proof of the Presentation Lemma



The Proof of the Presentation Lemma

- As usual, let $P_1 = (\Sigma_1, \delta_1, K_1)$, $P_2 = (\Sigma_2, \delta_2, K_2)$ and $\phi: \Sigma_1 \rightarrow \Sigma_2$ be given
- Let c be a theorem refutable using $\vdash_{P1}$, i.e. assume we are given the proof tree
- We show by structural induction that there is a corresponding (morphed) proof tree for (ϕ c) in $\vdash_{P2}$
- I will present only the most interesing cases

The Proof of the Presentation Lemma

- Two basic examples:

$$\textsc{Closed}\ \frac{}{A, \bot \vdash \bot} \quad \xrightarrow{\ \phi\ } \quad \frac{}{\phi\, A, \phi\, \bot \vdash \bot} = \frac{}{\phi\, A, \bot \vdash \bot}$$

$$\textsc{And}\ \frac{A, s \wedge t, s, t \vdash \bot}{A, s \wedge t \vdash \bot} \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ (s \wedge t), \phi\ s, \phi\ t \vdash \bot}{\phi\ A, \phi\ (s \wedge t) \vdash \bot}$$

$$= \frac{\phi\ A, (\phi\ s) \wedge (\phi\ t), \phi\ s, \phi\ t \vdash \bot}{\phi\ A, (\phi\ s) \wedge (\phi\ t) \vdash \bot}$$

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\phi} \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\phi} \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
  - α-equivalence: morphisms do not affect variables

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
  - α-equivalence: morphisms do not affect variables
  - β-reduction: $(\lambda x.t)\ t' \xrightarrow{\ \beta\ } t^x_{t'}$

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\phi} \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
    - α-equivalence: morphisms do not affect variables
    - β-reduction: $(\lambda x.t)\ t' \xrightarrow{\beta} t^x_{t'}$

$$(\lambda x.t)\ t' \xrightarrow{\phi} (\lambda x.\phi\ t)\ (\phi\ t')$$

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
  - α-equivalence: morphisms do not affect variables
  - β-reduction:

$$\begin{array}{ccc} (\lambda x.t)\ t' & \xrightarrow{\ \beta\ } & t^x_{t'} \\ \Big\downarrow \phi & & \\ (\lambda x.\phi\ t)\ (\phi\ t') & \xrightarrow{\ \beta\ } & (\phi\ t)^x_{\phi\ t'} \end{array}$$

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\phi} \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
    - α-equivalence: morphisms do not affect variables
    - β-reduction:

$$\begin{array}{ccccc}
(\lambda x.t)\ t' & \xrightarrow{\beta} & t^x_{t'} & & \\
\downarrow \phi & & & \searrow \phi & \\
(\lambda x.\phi\ t)\ (\phi\ t') & \xrightarrow{\beta} & (\phi\ t)^x_{\phi\ t'} & & \phi\ (t^x_{t'})
\end{array}$$

The Proof of the Presentation Lemma

Lemma: Let $\mathcal{P} = (\Sigma, \mathcal{K}, \delta)$ be a presentation as usual, s a well-typed Σ-Term and $\overline{\theta}$ a substitution on terms. Let ϕ be a signature morphism from Σ to some other signature. Then:

$$\phi\ (\overline{\theta}\ t) = \overline{\theta'}\ (\phi\ t)$$

where $\theta' = \phi \circ \theta$.

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot} \text{ where } s \sim_\lambda s' \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
  - α-equivalence: morphisms do not affect variables
  - β-reduction: $(\lambda x.t)\ t' \xrightarrow{\ \beta\ } t^x_{t'}$

$$\begin{array}{ccccc} (\lambda x.t)\ t' & \xrightarrow{\ \beta\ } & t^x_{t'} & & \\ \downarrow \phi & & & \searrow \phi & \\ (\lambda x.\phi\ t)\ (\phi\ t') & \xrightarrow{\ \beta\ } & (\phi\ t)^x_{\phi\ t'} & = & \phi\ (t^x_{t'}) \end{array}$$

The Proof of the Presentation Lemma

- The Lambda case

$$\textsc{Lambda}\ \frac{A, s, s' \vdash \bot}{A, s \vdash \bot}\ \text{where } s \sim_\lambda s' \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ s, \phi\ s' \vdash \bot}{\phi\ A, \phi\ s \vdash \bot}$$

- Claim: $\phi\ s \sim_\lambda \phi\ s'$ (i.e. we still have an instance of Lambda)
  - α-equivalence: morphisms do not affect variables
  - β-reduction:

$$\begin{array}{ccccc} (\lambda x.t)\ t' & \xrightarrow{\ \beta\ } & t^x_{t'} & & \\ \downarrow \phi & & & \searrow \phi & \\ (\lambda x.\phi\ t)\ (\phi\ t') & \xrightarrow{\ \beta\ } & (\phi\ t)^x_{\phi\ t'} & = & \phi\ (t^x_{t'}) \end{array}$$

  - η-reduction:

$$\begin{array}{ccc} \lambda x.t\ x & \xrightarrow{\ \eta\ } & t \\ \downarrow \phi & & \downarrow \phi \\ \lambda x.(\phi\ t)\ x & \xrightarrow{\ \eta\ } & \phi\ t \end{array}$$

The Proof of the Presentation Lemma

$$\textsc{Apply}{=}\ \frac{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t], C[\overline{\theta} s] \vdash \bot}{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t] \vdash \bot}$$

The Proof of the Presentation Lemma

$$\phi$$

$$\textsc{Apply}{=}\ \frac{A, \forall \overline{x^n}.s = t, C[\overline{\theta}t], C[\overline{\theta}s] \vdash \bot}{A, \forall \overline{x^n}.s = t, C[\overline{\theta}t] \vdash \bot} \qquad \frac{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta}t], \phi\, C[\overline{\theta}s] \vdash \bot}{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta}t] \vdash \bot}$$

The Proof of the Presentation Lemma

$$\phi$$

$$\text{APPLY}= \frac{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t], C[\overline{\theta} s] \vdash \bot}{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t] \vdash \bot} \qquad \frac{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t], \phi\, C[\overline{\theta} s] \vdash \bot}{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t] \vdash \bot}$$

Lemma: Let $\mathcal{P} = (\Sigma, \mathcal{K}, \delta)$ be a presentation as usual and $C[t]$ $\in$wff(Σ) some context with a term in its hole. Let ϕ be a signature morphism from Σ to some other signature. Then:

$$\phi\ (C[t]) = (\phi\ C)[(\phi t)]$$

The Proof of the Presentation Lemma

ϕ

$$\textsc{Apply}{=}\ \frac{A, \forall \overline{x^n}. s = t, C[\overline{\theta} t], C[\overline{\theta} s] \vdash \bot}{A, \forall \overline{x^n}. s = t, C[\overline{\theta} t] \vdash \bot} \qquad \frac{\phi\, A, \forall \overline{x^n}. (\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t], \phi\, C[\overline{\theta} s] \vdash \bot}{\phi\, A, \forall \overline{x^n}. (\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t] \vdash \bot}$$

$$\frac{\phi\ A, \forall \overline{x^n}. \phi\ s = \phi\ t, (\phi\ C)[\phi\ (\overline{\theta} t)], (\phi\ C)[\phi\ (\overline{\theta} s)] \vdash \bot}{\phi\ A, \forall \overline{x^n}. \phi\ s = \phi\ t, (\phi\ C)[\phi\ (\overline{\theta} t)] \vdash \bot}$$

The Proof of the Presentation Lemma

$$\phi$$

$$\textsc{Apply}{=}\ \frac{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t], C[\overline{\theta} s] \vdash \bot}{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t] \vdash \bot} \qquad \frac{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t], \phi\, C[\overline{\theta} s] \vdash \bot}{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t] \vdash \bot}$$

Lemma: Let $\mathcal{P} = (\Sigma, \mathcal{K}, \delta)$ be a presentation as usual, s a well-typed Σ-Term and $\overline{\theta}$ a substitution on terms. Let ϕ be a signature morphism from Σ to some other signature. Then:

$$\phi\ (\overline{\theta}\ t) = \overline{\theta'}\ (\phi\ t)$$

where $\theta' = \phi \circ \theta$.

The Proof of the Presentation Lemma

$$\textsc{Apply}{=}\ \frac{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t], C[\overline{\theta} s] \vdash \bot}{A, \forall \overline{x^n}.s = t, C[\overline{\theta} t] \vdash \bot} \quad \xrightarrow{\phi} \quad \frac{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t], \phi\, C[\overline{\theta} s] \vdash \bot}{\phi\, A, \forall \overline{x^n}.(\phi\, s) = (\phi\, t), \phi\, C[\overline{\theta} t] \vdash \bot}$$

$$\frac{\phi\ A, \forall \overline{x^n}.\phi\ s = \phi\ t, (\phi\ C)[\phi\ (\overline{\theta} t)], (\phi\ C)[\phi\ (\overline{\theta} s)] \vdash \bot}{\phi\ A, \forall \overline{x^n}.\phi\ s = \phi\ t, (\phi\ C)[\phi\ (\overline{\theta} t)] \vdash \bot}$$

$$\frac{\phi\ A, \forall \overline{x^n}.\phi\ s = \phi\ t, (\phi\ C)[\overline{\theta'}\ (\phi t)], (\phi\ C)[\overline{\theta'}\ (\phi s)] \vdash \bot}{\phi\ A, \forall \overline{x^n}.\phi\ s = \phi\ t, (\phi\ C)[\overline{\theta'}\ (\phi t)] \vdash \bot}$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}} \; \frac{A, k \vdash \bot}{A \vdash \bot} \text{ if } k \in \mathcal{K}$$

$$\phi \; A \vdash \bot$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}} \; \frac{A, k \vdash \bot}{A \vdash \bot} \text{ if } k \in \mathcal{K} \quad \xrightarrow{\;\phi\;} \quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}} \ \frac{A, k \vdash \bot}{A \vdash \bot} \text{ if } k \in \mathcal{K} \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

$$\phi\ A \vdash \bot$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}}\ \frac{A, k \vdash \bot}{A \vdash \bot}\ \text{if } k \in \mathcal{K}$$

$$\xrightarrow{\ \phi\ }\quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

$$\mathbf{XM}\ \frac{\phi\ A, \phi\ k \vee \neg(\phi\ k) \vdash \bot}{\phi\ A \vdash \bot}$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}}\ \frac{A, k \vdash \bot}{A \vdash \bot} \text{ if } k \in \mathcal{K} \quad \xrightarrow{\Phi} \quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

$$\textbf{XM}\ \dfrac{\textbf{Or}\ \dfrac{\phi\ A, \phi\ k \vee \neg(\phi\ k), \phi\ k \vdash \bot \qquad \phi\ A, \phi\ k \vee \neg(\phi\ k), \neg(\phi\ k) \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k) \vdash \bot}}{\phi\ A \vdash \bot}$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}} \ \frac{A, k \vdash \bot}{A \vdash \bot} \text{ if } k \in \mathcal{K} \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

$$\textsc{XM}\ \dfrac{\textsc{Or}\ \dfrac{\textsc{Weak}\ \dfrac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k), \phi\ k \vdash \bot} \qquad \dfrac{\neg(\phi\ k) \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k), \neg(\phi\ k) \vdash \bot}\ \textsc{Weak}}{\phi\ A, \phi\ k \vee \neg(\phi\ k) \vdash \bot}}{\phi\ A \vdash \bot}$$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}} \ \frac{A, k \vdash \bot}{A \vdash \bot} \text{ if } k \in \mathcal{K} \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

closed by IH

$$\textsc{XM}\ \dfrac{\textsc{Or}\ \dfrac{\textsc{Weak}\ \dfrac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k), \phi\ k \vdash \bot} \qquad \dfrac{\neg(\phi\ k) \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k), \neg(\phi\ k) \vdash \bot}\ \textsc{Weak}}{\phi\ A, \phi\ k \vee \neg(\phi\ k) \vdash \bot}}{\phi\ A \vdash \bot}$$

The Proof of the Presentation Lemma

Recall the Presentation Lemma:

If **$\phi(k)$** $\in$ **$P_2^\bullet$ for all k** $\in$ **K_1** and ($\phi(d) = \phi(\delta_1(d))$) $\in$ $P_2^\bullet$ for all d $\in$ Dom(δ) then ϕ is a theory morphism from $P_1^\bullet$ to $P_2^\bullet$.

The Proof of the Presentation Lemma

Recall the Presentation Lemma:

If **$\phi(k)$** $\in$ **$P_2^\bullet$ for all k** $\in$ **K_1** and $(\phi(d) = \phi(\delta_1(d))) \in P_2^\bullet$ for all $d \in Dom(\delta)$ then ϕ is a theory morphism from $P_1^\bullet$ to $P_2^\bullet$.

=> $\phi(k)$ is refutable in P_2
=> There is a closed proof tree for $\neg\phi(k) \vdash \bot$

The Proof of the Presentation Lemma

$$\textsc{Axiom}_{\mathcal{P}}\ \frac{A, k \vdash \bot}{A \vdash \bot}\ \text{if } k \in \mathcal{K} \quad \xrightarrow{\ \phi\ } \quad \frac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A \vdash \bot}$$

closed by IH

closed by precondition

$$\textsc{XM}\ \dfrac{\textsc{Or}\ \dfrac{\textsc{Weak}\ \dfrac{\phi\ A, \phi\ k \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k), \phi\ k \vdash \bot} \qquad \dfrac{\neg(\phi\ k) \vdash \bot}{\phi\ A, \phi\ k \vee \neg(\phi\ k), \neg(\phi\ k) \vdash \bot}\ \textsc{Weak}}{\phi\ A, \phi\ k \vee \neg(\phi\ k) \vdash \bot}}{\phi\ A \vdash \bot}$$

The Proof of the Presentation Lemma

$$\textsc{ApplyDef}_{\mathcal{P}} \; \frac{A, C[c], C[\delta \; c] \vdash \bot}{A, C[c] \vdash \bot} \text{ if } c \in Dom(\delta)$$

$$\phi \; A, (\phi \; C)[\phi \; c] \vdash \bot$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta) \quad \longrightarrow \quad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot} \text{ if } c \in Dom(\delta) \qquad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

$$\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}} \ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot} \text{ if } c \in Dom(\delta) \qquad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

$$\textbf{XM} \ \frac{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vee \neg(\phi\ c = \phi\ (\delta\ c))}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta) \qquad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

$$\textbf{XM}\ \cfrac{\textsc{Or+Weak}\ \cfrac{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vdash \bot \qquad \neg(\phi\ c = \phi\ (\delta\ c)) \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \lor \neg(\phi\ c = \phi\ (\delta\ c))}}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta) \qquad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

$$\textbf{XM}\ \dfrac{\textbf{Or+Weak}\ \dfrac{\textbf{Apply=+Weak}\ \dfrac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vdash \bot} \qquad \neg(\phi\ c = \phi\ (\delta\ c)) \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vee \neg(\phi\ c = \phi\ (\delta\ c))}}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}}\ \dfrac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta) \qquad \dfrac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

closed by IH

$$\textbf{XM}\ \dfrac{\textbf{Or+Weak}\ \dfrac{\textbf{Apply=+Weak}\ \dfrac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vdash \bot} \qquad \neg(\phi\ c = \phi\ (\delta\ c)) \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vee \neg(\phi\ c = \phi\ (\delta\ c))}}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

The Proof of the Presentation Lemma

ϕ + Lemma

$$\textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta) \qquad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

closed by IH

Again, recall the Presentation Lemma:

If ϕ(k) ∈ $P_2^\bullet$ for all k ∈ K_1 and **(ϕ(d) = ϕ(δ_1(d))) ∈ $P_2^\bullet$ for all d ∈ Dom(δ)** then ϕ is a theory morphism from $P_1^\bullet$ to $P_2^\bullet$.

$$\textsc{XM}\ \frac{\textsc{Or+Weak}\ \dfrac{\textsc{Apply=+Weak}\ \dfrac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vdash \bot} \qquad \neg(\phi\ c = \phi\ (\delta\ c)) \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vee \neg(\phi\ c = \phi\ (\delta\ c))}}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

The Proof of the Presentation Lemma

Again, recall the Presentation Lemma:

If $\phi(k) \in P_2^\bullet$ for all $k \in K_1$ and **($\phi(d) = \phi(\delta_1(d))$) $\in$ $P_2^\bullet$ for all d $\in$ Dom(δ)** then ϕ is a theory morphism from $P_1^\bullet$ to $P_2^\bullet$.

=> ($\phi(d) = \phi(\delta_1(d))$) is refutable in P_2

=> There is a closed proof tree for $\neg(\phi(d) = \phi(\delta_1(d))) \vdash \bot$

The Proof of the Presentation Lemma

ϕ + Lemma

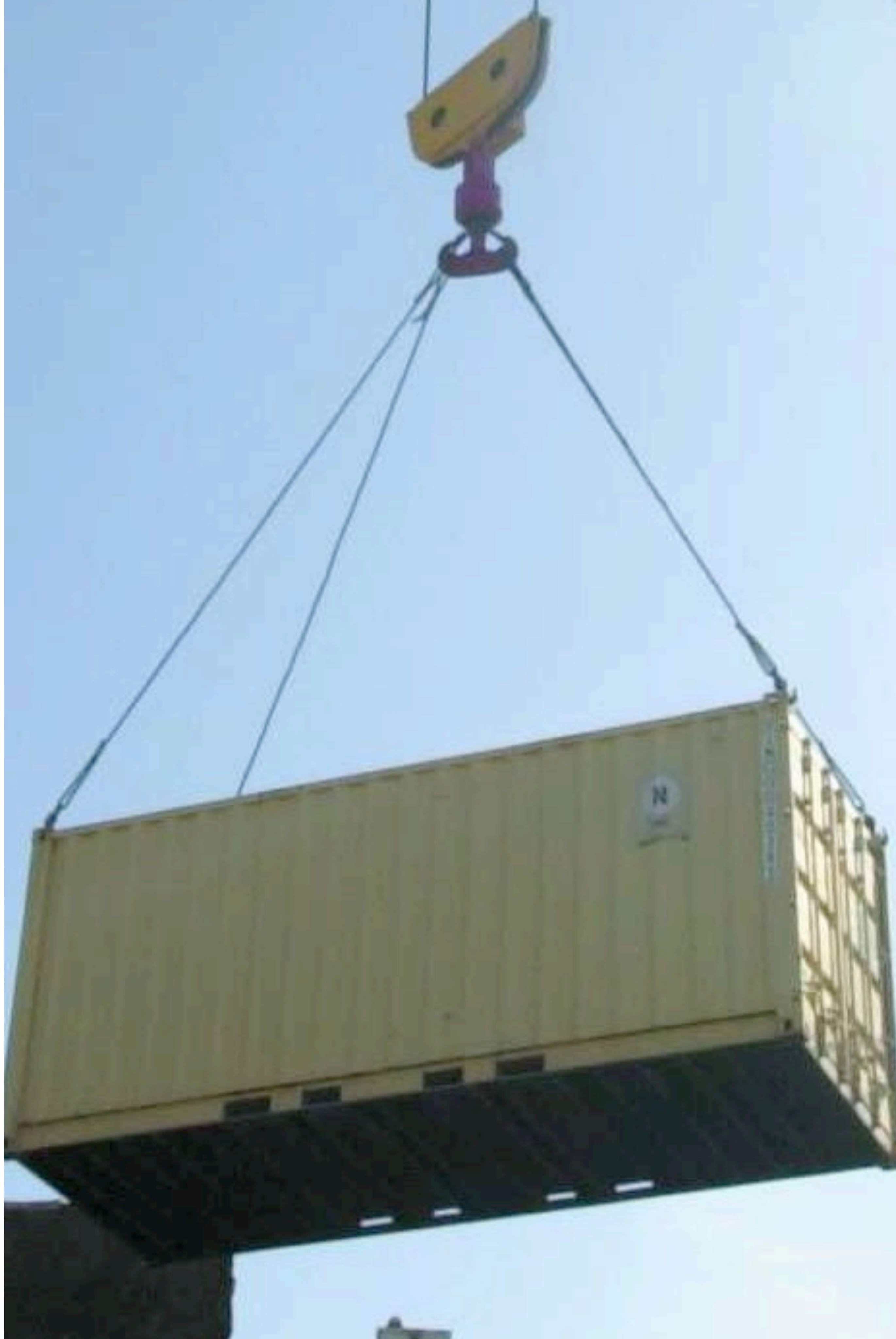
$$\textsc{ApplyDef}_{\mathcal{P}}\ \frac{A, C[c], C[\delta\ c] \vdash \bot}{A, C[c] \vdash \bot}\ \text{if } c \in Dom(\delta) \qquad \frac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

closed by IH

closed by precondition

$$\textbf{XM}\ \dfrac{\textbf{Or+Weak}\ \dfrac{\textbf{Apply=+Weak}\ \dfrac{\phi\ A, (\phi\ C)[\phi\ c], (\phi\ C)[\phi\ (\delta\ c)] \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vdash \bot} \qquad \neg(\phi\ c = \phi\ (\delta\ c)) \vdash \bot}{\phi\ A, (\phi\ C)[\phi\ c], \phi\ c = \phi\ (\delta\ c) \vee \neg(\phi\ c = \phi\ (\delta\ c))}}{\phi\ A, (\phi\ C)[\phi\ c] \vdash \bot}$$

Imports



From Morphisms to Imports

- Using only an implementation of pure morphisms is not very realistic:

- Assume, we want to reuse sort I in Presentation 2. Using morphisms, this would work as follows:

From Morphisms to Imports

- Using only an implementation of pure morphisms is not very realistic:

Presentation 1

```
sort I
```

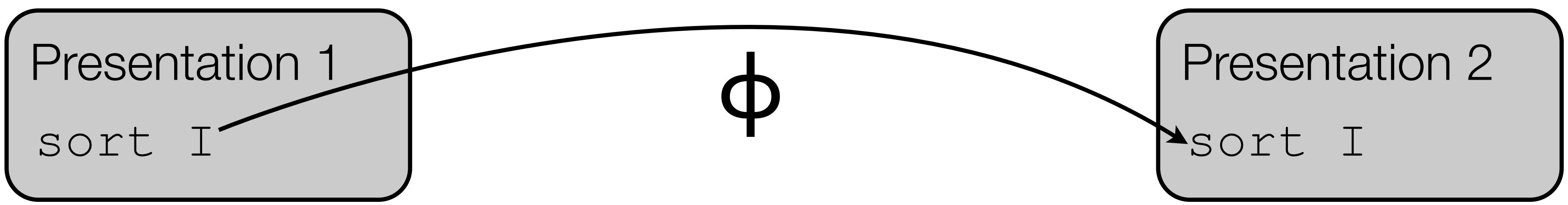
Presentation 2

```
sort I
```

- Assume, we want to reuse sort I in Presentation 2. Using morphisms, this would work as follows:
  - Define a sort I in Presentation 2

From Morphisms to Imports

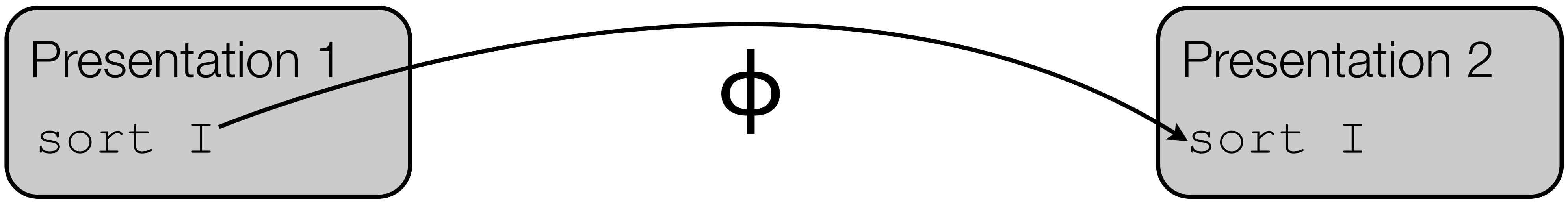
- Using only an implementation of pure morphisms is not very realistic:



- Assume, we want to reuse sort I in Presentation 2. Using morphisms, this would work as follows:
    - Define a sort I in Presentation 2
    - Map sort I of Presentation 1 to sort I of Presentation 2

From Morphisms to Imports

- Using only an implementation of pure morphisms is not very realistic:



- Assume, we want to reuse sort I in Presentation 2. Using morphisms, this would work as follows:
    - Define a sort I in Presentation 2
    - Map sort I of Presentation 1 to sort I of Presentation 2
- Quite useless, similar with constants, definitions...

From Morphisms to Imports ctd

- We need a possibility to define a presentation and morph another presentation at the same time, so called ***imports***

- Imports are more powerful practical counterparts to the theory of morphisms

Presentation 1

```
sort I
term union = \C, D:I B.\x:I.(C x) | (D x)
```

Presentation 2

```
import "Presentation 1"
end
sort M
...
```

- Implicitly defines sort I and definition union and applies identity morphism

More complex import

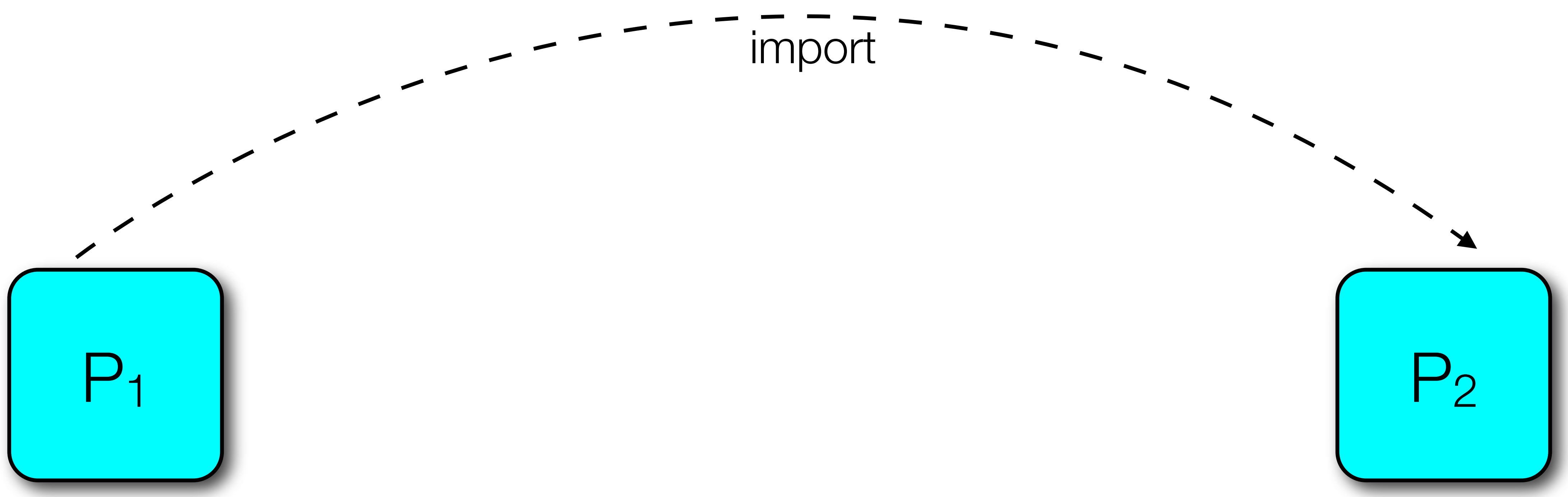
Presentation 1

```
sort I
term union = \C, D:I B.\x:I.(C x) | (D x)
```

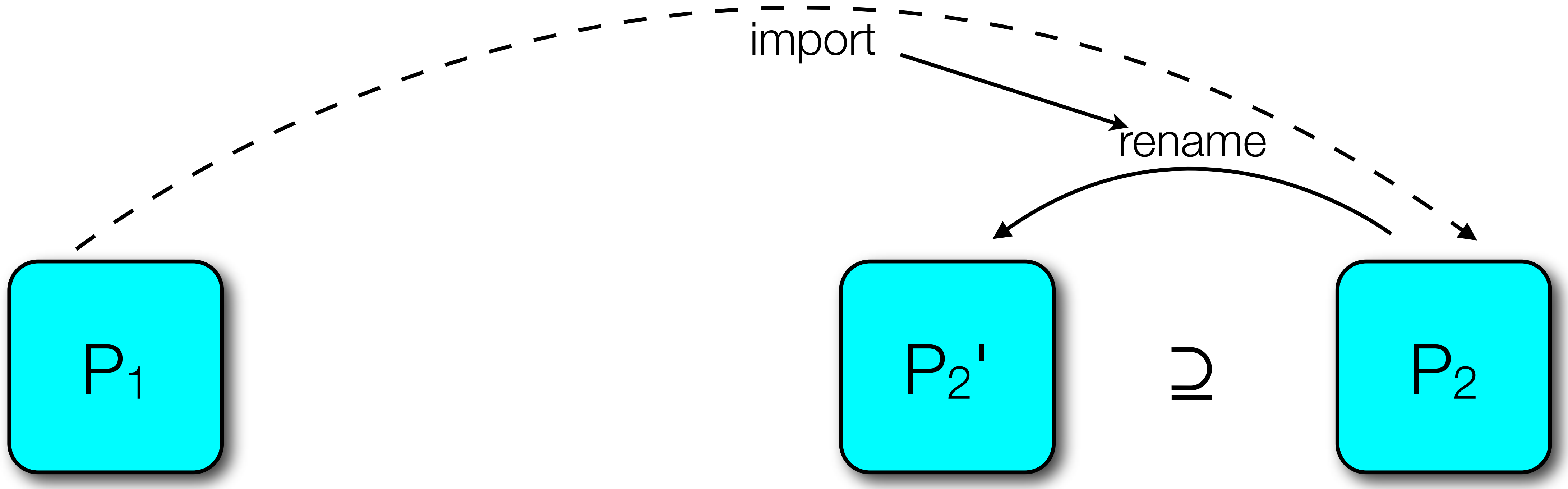
Presentation 2

```
sort V; // vertices
var v1, v2, v3: V;
const E: V V B; // edges
axiom !v1 v2. (E v1 v2) -> (E v2 v1); // undirected graph

import "Presentation 1"
    morph sort I = V // morphs sort I to sort V
    rename term union union_vertices // redefines union, renames it to union_vertices
                                     // and applies morphism (union->union_vertices)
end

claim !v1, v2, v3. (E v1 v3) ->
                   (union_vertices (E v1) (E v2)) v3
```

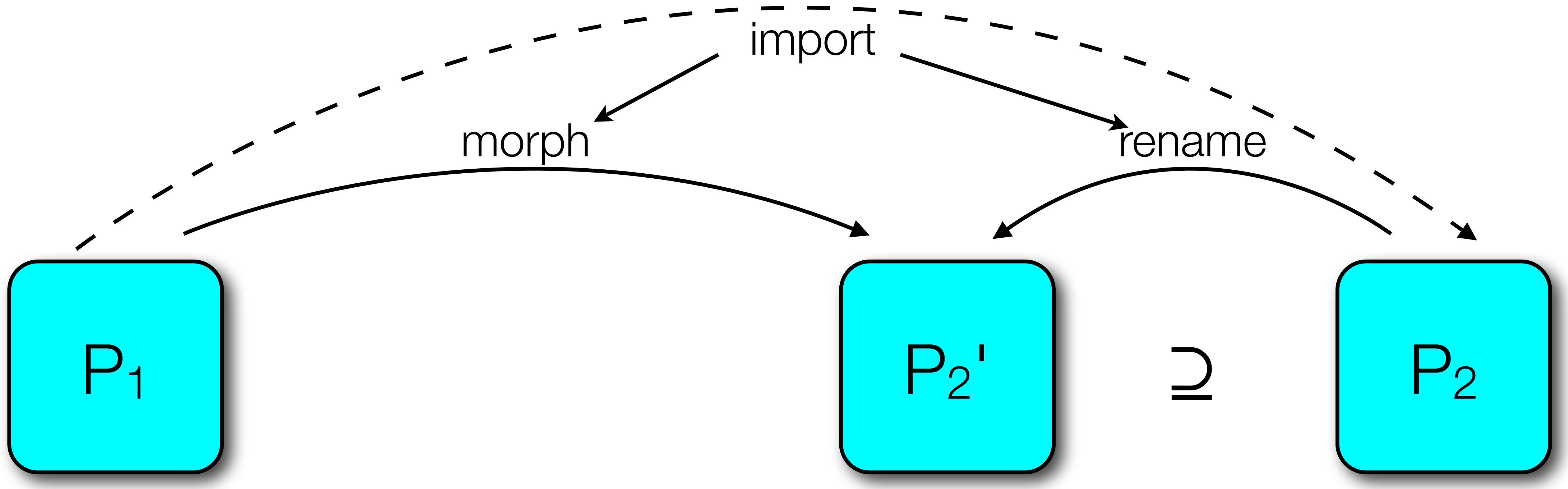
How imports work



How imports work



How imports work



Imports and the Presentation Lemma

- What about the obligations for a theory morphisms?
  - Morphed knowns must be provable
  - (Morphed constant = morphed definition) must be provable
- When using `rename` for knowns or definitions (i.e. if these elements are added to the target presentation), these proofs become trivial
- Otherwise: The corresponding obligation becomes a claim in the new presentation and has to be proven by the user

Default Import Mode

Presentation 1

```
sort I
term union = \C, D:I B.\x:I.(C x) | (D x)
```

Presentation 2

```
sort I
import "Presentation 1"
end
```

- Does not work, sort I already exists in presentation 2
- => if nothing is specified (e.g. by `rename` or `morph`), the system checks
  - if the corresponding element already exists => only identity morphism
  - if not => the element is added to the presentation => identity morphism
  - if the corresponding element already exists but term/type does not match => error

The Danger of Imports

Natural Numbers

```
sort N // natural numbers
const 0:N // zero
const S:N N // successor function
axiom !x:N, y:N. (S x = S y) -> x = y // injectivity of S
axiom !x:N. S x != 0 // successor of a number is never zero
axiom !p:N B. p 0 & (!x:N. p x -> p (S x)) -> !x:N. p x // induction axiom
```

- We morph N to N B, 0 to {0}
- We morph S to a function, which, given a subset, adds the lowest number to this set which is not contained in it, e.g. {1, 2, 3, 5, 6} -> {1, 2, 3, 4, 5, 6}

Subsets of Natural Numbers

```
sort N // natural numbers
// here begins the import
axiom !x:N B, y:N B. (S x = S y) -> x = y
axiom !x:N B. S x != {0 }
axiom !p:N B. p 0 & (!x:N B. p x -> p (S x)) -> !x:N B. p x
```

- Consider the empty set...

Implementation

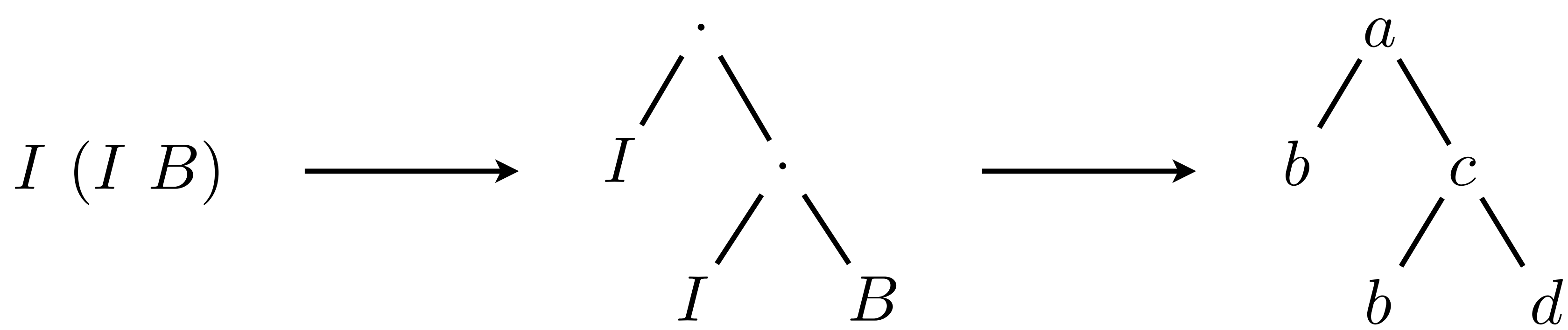
Some Statistics

- Implementation in PHP / HTML / Javascript
- PostgreSQL as database
- About 12000 lines of pure code (i.e. without comments etc)
- Following tests performed on a Fedora Linux in a XEN virtual machine running on an AMD Athlon 64 X2 5600+ Dual Core with 2 GB DDR2 RAM and a 400GB SATAII hard disk

Performance Problems

- Test case: A chain of 300 presentation imports, i.e. a presentation which imports a presentation which imports a presentation...
- Each import adds only one lemma => about 300 axioms
- Loading took over one minute
  - Reason: Thousands (!) of database queries
  - Solution: see next slides
- Morphing needed over 160 MB memory
  - Reason: Everything was copied when morphed.
  - Solution: Only copy things which are really affected by a morphism => Memory consumption went down to 130 MB

A Datastructure for Storing Trees



- Pointer Structure:

Nodes

a
b
c
d

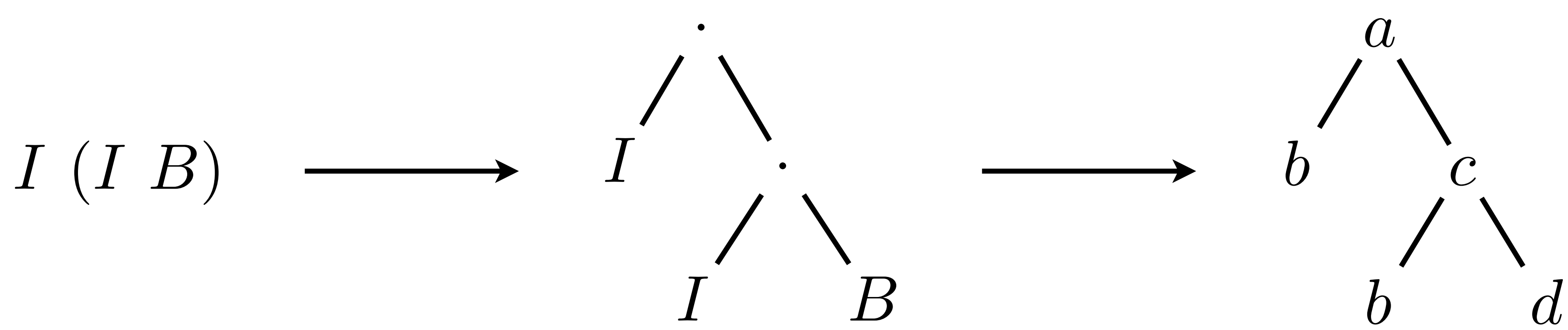
Children

a	b
a	c
c	b
c	d

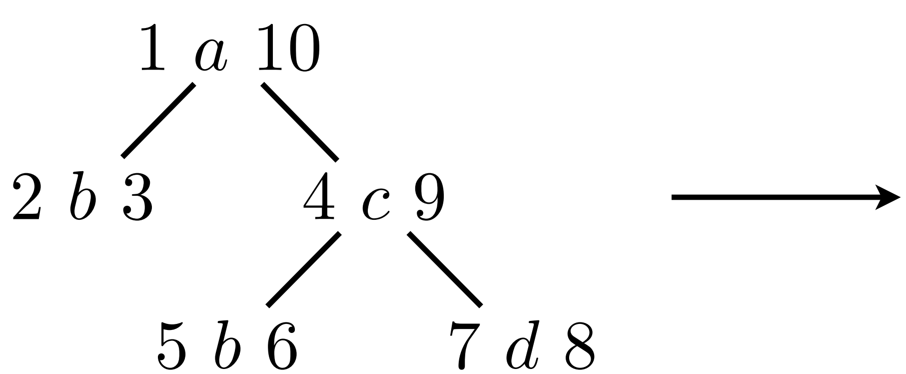
- 2n queries to load, 2n queries to store (worst case)

- Redundancies can be used to reduce storage/number of operations

A Datastructure for Storing Trees ctd



- Nested Set Structure: Depth first search



1	10	a
2	3	b
4	9	c
5	6	b
7	8	d

- n queries to store
- 1 query to load
- Redundancies can only rarely be used

A Datastructure for Storing Trees ctd

- Test case: Random, full binary tree with 2047 nodes
- 3 different leafes => lot of redundancies (advantage for pointer structure)
- Storage needed:
  - Pointer structure, optimized for binary trees: 343 rows
  - Nested Set: 2047 rows
- Time needed for loading:
  - Pointer structure: 0.28 seconds
  - Nested Set: 0.12 seconds

Optimization Results

- Remember: Before optimization:
  - Loading of 300 imports took more than a second
- Implemented optimizations:
  - Nested Set structure for terms and types
  - Union Queries (not explained here)
- Result: Loading of 300 imports takes about 10 seconds now

Demo time!

Future Work

- Implementation of proofs as a tree of presentations
- Possibility to search for presentation elements by name, term and type
- Implementation of a syntax for imports in Jitpro
- Restricted morphisms, e.g. N is mapped to N B such that it is not the empty set

Thank you!

Enjoy your week ;-)

References

- Gert Smolka, Chad E. Brown: *Introduction to Computational Logic - Lecture Notes SS 2008*. 2008.
- Chad E. Brown: *Jitpro, A JavaScript Interactive Higher-Order Tableau Prover*.
- R.M. Burstall, J.A. Goguen: *Putting theories together to make specifications*. In Proceedings of the 5th International Joint Conference on Artificial Intelligence, 1045–1058, 1977.
- J.A. Goguen, R.M. Burstall: *Institutions: Abstract Model Theory for Specification and Programming*. Journal of the ACM, Volume 39, 95–146, 1992.
- William M. Farmer, Joshua D. Guttman, F. Javier Thayer: *Little Theories*. In Proceedings of the 11th International Conference on Automated Deduction, 567–581, 1992.
- Michael J. Kamfonas: *Recursive Hierarchies: The Relational Taboo!*. The Relational Journal, 1992.