Correct/Incorrect?

Does the following Prolog program produce for every lambda-term the correct type?

```
type Gamma (var X) A :- member (pair X A) Gamma.

type Gamma (app M N) B :- type Gamma M (arrow A B),
                          type Gamma N A.

type Gamma (lam X M) (arrow A B) :-
                          type (pair X A)::Gamma M B.

member A A::Tail.

member A B::Tail :- member A Tail.
```

Nominal Techniques Course

Wednesday-Lecture

Christian Urban

University of Cambridge

Recap from Yesterday

Nominal Logic has the following weak (in the good sense) induction principle for lambda-terms:

$$\frac{\begin{array}{l} (\forall a : Var)\ \varphi(var(a),\vec{x}) \\ (\forall t_1, t_2 : Trm)\ \varphi(t_1,\vec{x}) \wedge \varphi(t_2,\vec{x}) \\ \qquad\qquad\qquad\qquad \Rightarrow \varphi(app(t_1,t_2),\vec{x}) \\ (\exists a : Var)\ a \# \vec{x} \wedge (\forall t : Trm)\ \varphi(t,\vec{x}) \\ \qquad\qquad\qquad\qquad \Rightarrow \varphi(lam(a.t),\vec{x}) \end{array}}{(\forall t : Trm)\ \varphi(t,\vec{x})}$$

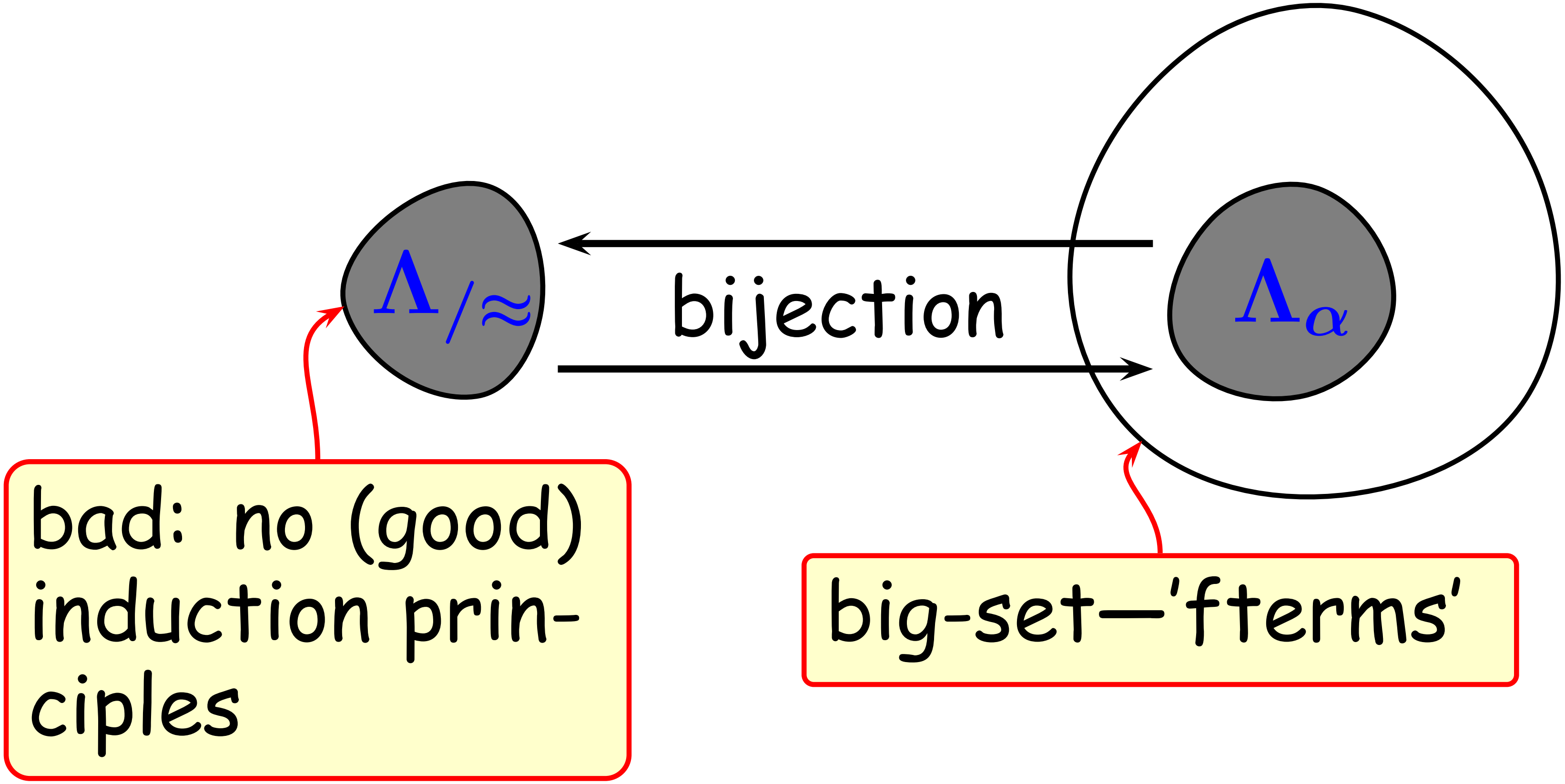
It asks that for every term there exists a fresh atom.

$$(\forall x : S)(\exists a : A)\ \ a \# x$$

Are such principles justified? Answer in today's lecture.

General Outline

We shall define a '**big-set**' and then carve out a '**small-set**', Λ_α, that is bijective with $\Lambda_{/\approx}$.



bad: no (good) induction principles

big-set—'fterms'

General Outline

We shall define a '**big-set**' and then carve out a '**small-set**', Λ_α, that is bijective with $\Lambda_{/\approx}$.

Caveat: The lambda-calculus is now more than 60 years old and people have tried for a long time to find a simple solution for the problem with binders. This means what I present next is necessarily a bit complicated. It get's simple again on Friday. ;o)

Small Dictionary

- Λ set of (raw)-lambda-terms
- $\Lambda_{/\approx}$ set of α-equated lambda-terms (not inductively defined)
- big-set also $Ftrm$ (inductively defined)
- small-set also Λ_{α} (subset of big-set, inductively defined, in bijection with $\Lambda_{/\approx}$)

Big-Set

Naïve attempt for big-set

$$
\begin{array}{llll}
\boldsymbol{Ftrm} ::= & & & \\
 & \mathsf{am} : \boldsymbol{Atom} & & \text{'atoms'} \\
 | & \mathsf{pr} : \boldsymbol{Ftrm} \times \boldsymbol{Ftrm} & & \text{'pairs'} \\
 | & \mathsf{se} : \boldsymbol{Ftrm}\ \boldsymbol{Set} & & \text{'}\alpha\text{-eq-cl'}
\end{array}
$$

Big-Set

Naïve attempt for big-set

$$
\begin{array}{llll}
\boldsymbol{Ftrm} ::= & & & \\
 & \textsf{am} & : \boldsymbol{Atom} & \text{'atoms'} \\
 | & \textsf{pr} & : \boldsymbol{Ftrm} \times \boldsymbol{Ftrm} & \text{'pairs'} \\
 | & \textsf{se} & : \boldsymbol{Ftrm\ Set} & \text{'}\alpha\text{-eq-cl'}
\end{array}
$$

trick: encode the α-equivalence class as the set of lambda-terms

$$[t]_\alpha \overset{\text{def}}{=} \{t' \mid t \approx t'\}$$

Big-Set

Better attempt for big-set

$$
\begin{array}{rlll}
\mathit{Ftrm} ::= & \mathsf{er} & & \text{'error'} \\
\mid & \mathsf{am} & : \mathit{Atom} & \text{'atoms'} \\
\mid & \mathsf{pr} & : \mathit{Ftrm} \times \mathit{Ftrm} & \text{'pairs'} \\
\mid & \mathsf{se} & : \mathit{Atom} \longrightarrow \mathit{Ftrm} & \text{'}\alpha\text{-eq-cl'}
\end{array}
$$

same idea, but encoding with (partial) functions, along the lines:

"if $t' \in [t]_\alpha$ then yes else er"

You Could Guessed It: Permutation for Big-Set

Starting from the permutation operation for atoms, we want to permute all **free** atoms in fterms:

$$
\begin{array}{lcl}
\pi\bullet\mathsf{er} & \overset{\mathsf{def}}{=} & \mathsf{er} \\
\pi\bullet\mathsf{am}(a) & \overset{\mathsf{def}}{=} & \mathsf{am}(\pi\bullet a) \\
\pi\bullet\mathsf{pr}(t_1, t_2) & \overset{\mathsf{def}}{=} & \mathsf{pr}(\pi\bullet t_1, \pi\bullet t_2) \\
\pi\bullet\mathsf{se}(fn) & \overset{\mathsf{def}}{=} & \mathsf{se}(\lambda a.\pi\bullet(fn(\pi^{-1}\bullet a)))
\end{array}
$$

You Could Guessed It:

Ok, slowly: fn is a function $Atom \rightarrow Ftrm$

$$fn = \lambda a.(fn\ a)$$

So we should have

$$\pi\bullet fn = \pi\bullet\lambda a.(fn\ a)$$

We want to permute all free atoms in fn ($= \lambda a.(fn\ a)$—a is clearly **not** free). Therefore

$$\lambda a.\pi\bullet(fn\ a)$$

is wrong, as it will also permute a (wherever it ends up). However, if we substitute $\pi^{-1}\bullet a$ first, then the π that is too much will go away.

$$\pi\bullet \mathsf{se}(fn) \overset{\mathsf{def}}{=} \mathsf{se}(\lambda a.\pi\bullet(fn(\pi^{-1}\bullet a)))$$

Properties of this Permutation Operation

$$
\begin{aligned}
\pi\bullet \mathsf{er} &\overset{\text{def}}{=} \mathsf{er}\\
\pi\bullet \mathsf{am}(a) &\overset{\text{def}}{=} \mathsf{am}(\pi\bullet a)\\
\pi\bullet \mathsf{pr}(t_1,t_2) &\overset{\text{def}}{=} \mathsf{pr}(\pi\bullet t_1,\pi\bullet t_2)\\
\pi\bullet \mathsf{se}(fn) &\overset{\text{def}}{=} \mathsf{se}(\lambda a.\pi\bullet(fn(\pi^{-1}\bullet a)))
\end{aligned}
$$

- $[]\bullet t = t$
- $(\pi_1 @ \pi_2)\bullet t = \pi_1\bullet(\pi_2\bullet t)$
- $ds(\pi_1,\pi_2)=\varnothing$ implies $\pi_1\bullet t = \pi_2\bullet t$

Properties of this Permutation Operation

$\pi\bullet$er $\overset{\text{def}}{=}$ er

$\pi\bullet$am(a) $\overset{\text{def}}{=}$ am$(\pi\bullet a)$

$\pi\bullet$pr

$\pi\bullet$se

If a type (set) satisfies these three properties, then we call it a **permutation type**. So $Ftrm$'s are a permutation type—or short $PType$.

- $[]\bullet t = t$
- $(\pi_1 @ \pi_2)\bullet t = \pi_1\bullet(\pi_2\bullet t)$
- $ds(\pi_1, \pi_2) = \varnothing$ implies $\pi_1\bullet t = \pi_2\bullet t$

Abstract Properties

If a type satisfies

- $[]\bullet t = t$
- $(\pi_1 @ \pi_2)\bullet t = \pi_1\bullet(\pi_2\bullet t)$
- $ds(\pi_1, \pi_2) = \varnothing$ implies $\pi_1\bullet t = \pi_2\bullet t$

we can prove (independent of what the type looks like)

- $(a\,a)\bullet t = t$
- $\pi^{-1}\bullet(\pi\bullet t) = t$
- $\pi\bullet t_1 = t_2$ iff $t_1 = \pi^{-1}\bullet t_2$
- $t \in X$ iff $\pi\bullet t \in \pi\bullet X$

where $\pi\bullet X \overset{\text{def}}{=} \{\pi\bullet t \mid t \in X\}$

BTW: Where Do Atoms Come From?

We assume a countable infinite set of atoms. Countable infinite is important!

For example, the natural numbers would do—just we do not write them as numbers, rather as

$$a, b, c, \ldots$$

The only property we are interested in is that there are countably infinite many atoms: no hidden games with de-Bruijn indices.

SUPPORT!!!

Once we have a permutation operation for a type, we can define the notion of support (a set of atoms):

$$\text{supp} : \mathit{PType} \rightarrow \mathit{Atom\ Set}$$

$$\text{supp}(x) \overset{\text{def}}{=} \{a \mid \text{infinite}\,\{b \mid (a\,b)\bullet x \neq x\}\}$$

In words: all atoms a where the set

$$\{b \mid (a\,b)\bullet x \neq x\}$$

is infinite (each swapping $(a\,b)$ needs to change something “syntactically” in x).

Digression: λ-Calculus

The (raw) lambda-calculus is a ptype.

$$
\begin{aligned}
\pi\bullet a &\overset{\text{def}}{=} \begin{cases} a_1 & \text{if } \pi\bullet a = a_2 \\ a_2 & \text{if } \pi\bullet a = a_1 \\ \pi\bullet a & \text{otherwise} \end{cases} \\
\pi\bullet t_1\, t_2 &\overset{\text{def}}{=} (\pi\bullet t_1)(\pi\bullet t_2) \\
\pi\bullet \lambda a.t &\overset{\text{def}}{=} \lambda(\pi\bullet a).(\pi\bullet t)
\end{aligned}
$$

- $[]\bullet t = t$
- $(\pi_1 @ \pi_2)\bullet t = \pi_1\bullet(\pi_2\bullet t)$
- $ds(\pi_1, \pi_2) = \varnothing$ implies $\pi_1\bullet t = \pi_2\bullet t$

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

$$a\text{:} \quad (a\,?)\bullet c \neq c$$

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

a: $(a\,?)\bullet c \neq c$ no

b: $(b\,?)\bullet c \neq c$

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

a:	$(a\,?)\bullet c \neq c$	no
b:	$(b\,?)\bullet c \neq c$	no
c:	$(c\,?)\bullet c \neq c$	

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b){\bullet}c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

a:	$(a\,?){\bullet}c \neq c$	no
b:	$(b\,?){\bullet}c \neq c$	no
c:	$(c\,?){\bullet}c \neq c$	yes
d:	$(d\,?){\bullet}c \neq c$	

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

a:	$(a\,?)\bullet c \neq c$	no
b:	$(b\,?)\bullet c \neq c$	no
c:	$(c\,?)\bullet c \neq c$	yes
d:	$(d\,?)\bullet c \neq c$	no
	$\vdots$	no

Support of an Atom

What is the support of the atom c?

$$\mathsf{supp}(c) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet c \neq c\}\}$$

Let's check the (infinitely many) atoms one by one:

So $\mathsf{supp}(c) = \{c\}$

a:	$(a\,?)\bullet c \neq c$	no
b:	$(b\,?)\bullet c \neq c$	no
c:	$(c\,?)\bullet c \neq c$	yes
d:	$(d\,?)\bullet c \neq c$	no
	$\vdots$	no

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid ((a\,b)\bullet t_1)\;((a\,b)\bullet t_2) \neq t_1\,t_2\}\}$$

Support of an Application

$$\mathsf{supp}(t_1\, t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\, \{b \mid (a\, b)\bullet t_1\, t_2 \neq t_1\, t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid ((a\, b)\bullet t_1)\, ((a\, b)\bullet t_2) \neq t_1\, t_2\}\}$$

We know

$$t_1\, t_2 = s_1\, s_2 \;\text{ iff }\; t_1 = s_1 \wedge t_2 = s_2$$

hence

$$t_1\, t_2 \neq s_1\, s_2 \;\text{ iff }\; t_1 \neq s_1 \vee t_2 \neq s_2$$

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid ((a\,b)\bullet t_1)\;((a\,b)\bullet t_2) \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1 \vee (a\,b)\bullet t_2 \neq t_2\}\}$$

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid ((a\,b)\bullet t_1)\,((a\,b)\bullet t_2) \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1 \vee (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}(\{b \mid (a\,b)\bullet t_1 \neq t_1\} \cup \{b \mid (a\,b)\bullet t_2 \neq t_2\})\}$$

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

$$\{a \,|\, \mathsf{inf}\{b \,|\, ((a\,b)\bullet t_1)\;((a\,b)\bullet t_2) \neq t_1\,t_2\}\}$$

$$\{a \,|\, \mathsf{inf}\{b \,|\, (a\,b)\bullet t_1 \neq t_1 \vee (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\{a \,|\, \mathsf{inf}(\{b \,|\, (a\,b)\bullet t_1 \neq t_1\}\cup\{b \,|\, (a\,b)\bullet t_2 \neq t_2\})\}$$

$$\{a \,|\, \mathsf{inf}\{b \,|\, (a\,b)\bullet t_1 \neq t_1\}\vee \mathsf{inf}\{b \,|\, (a\,b)\bullet t_2 \neq t_2\}\}$$

Support of an Application

$$\mathsf{supp}(t_1\, t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\, b)\bullet t_1\, t_2 \neq t_1\, t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid ((a\, b)\bullet t_1)\ ((a\, b)\bullet t_2) \neq t_1\, t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\, b)\bullet t_1 \neq t_1 \vee (a\, b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}(\{b \mid (a\, b)\bullet t_1 \neq t_1\} \cup \{b \mid (a\, b)\bullet t_2 \neq t_2\})\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\, b)\bullet t_1 \neq t_1\} \vee \mathsf{inf}\{b \mid (a\, b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\, b)\bullet t_1 \neq t_1\}\} \cup \{a \mid \mathsf{inf}\{b \mid (a\, b)\bullet t_2 \neq t_2\}\}$$

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid ((a\,b)\bullet t_1)\;((a\,b)\bullet t_2) \neq t_1\,t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1 \vee (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}(\{b \mid (a\,b)\bullet t_1 \neq t_1\} \cup \{b \mid (a\,b)\bullet t_2 \neq t_2\})\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1\} \vee \mathsf{inf}\{b \mid (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1\}\} \cup \{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\mathsf{supp}(t_1) \qquad\qquad \cup \qquad\qquad \mathsf{supp}(t_2)$$

Support of an Application

$$\mathsf{supp}(t_1\,t_2) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t_1\,t_2 \neq t_1\,t_2\}\}$$

$\{a \mid \mathsf{inf}\{b \mid$

So $\mathsf{supp}(t_1\,t_2) = \mathsf{supp}(t_1) \cup \mathsf{supp}(t_2)$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1 \vee (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}(\{b \mid (a\,b)\bullet t_1 \neq t_1\} \cup \{b \mid (a\,b)\bullet t_2 \neq t_2\})\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1\} \vee \mathsf{inf}\{b \mid (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_1 \neq t_1\}\} \cup \{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet t_2 \neq t_2\}\}$$

$$\mathsf{supp}(t_1) \qquad\qquad \cup \qquad\qquad \mathsf{supp}(t_2)$$

Support of an Abstraction

$$\mathsf{supp}(\lambda c.t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet\lambda c.t \neq \lambda c.t\}\}$$

Support of an Abstraction

$$\mathsf{supp}(\lambda c.t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet\lambda c.t \neq \lambda c.t\}\}$$

We mean here 'syntactic' (in)-equality, **not** α-(in)-equality.

Support of an Abstraction

$$\mathsf{supp}(\lambda c.t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b) \bullet \lambda c.t \neq \lambda c.t\}\}$$

So $\boxed{\mathsf{supp}(\lambda c.t) = \mathsf{supp}(t) \cup \{c\}}$

Support for λ-Terms

$$\mathsf{supp}(t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t \neq t\}\}$$

- $\mathsf{supp}(c) = \{c\}$
- $\mathsf{supp}(t_1 t_2) = \mathsf{supp}(t_1) \cup \mathsf{supp}(t_1)$
- $\mathsf{supp}(\lambda c.t) = \mathsf{supp}(t) \cup \{c\}$

Support for λ-Terms

$$\text{supp}(t) \overset{\text{def}}{=} \{a \mid \text{infinite}\,\{b \mid (a\,b)\bullet t \neq t\}\}$$

- $\text{supp}(c) = \{c\}$
- $\text{supp}(t_1 t_2) = \text{supp}(t_1) \cup \text{supp}(t_1)$
- $\text{supp}(\lambda c.t) = \text{supp}(t) \cup \{c\}$

$\text{supp}(t) = \text{occurs}(t)$ (for lambda-terms)

- $\text{occurs}(c) \overset{\text{def}}{=} \{c\}$
- $\text{occurs}(t_1 t_2) \overset{\text{def}}{=} \text{occurs}(t_1) \cup \text{occurs}(t_1)$
- $\text{occurs}(\lambda c.t) \overset{\text{def}}{=} \text{occurs}(t) \cup \{c\}$

A Variant

$$\mathsf{supp}'(t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t \neq t\}\}$$

A Variant

$$\mathsf{supp}'(t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t \not\approx t\}\}$$

A Variant

$$\mathsf{supp}'(t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t \not\approx t\}\}$$

$$\begin{aligned}\mathsf{supp}'(\lambda c.c) &= \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet \lambda c.c \not\approx \lambda c.c\}\} \\ &= \varnothing\end{aligned}$$

A Variant

$$\mathsf{supp}'(t) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet t \not\approx t\}\}$$

$$\begin{aligned}\mathsf{supp}'(\lambda c.c) &= \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet \lambda c.c \not\approx \lambda c.c\}\}\\ &= \varnothing\end{aligned}$$

$$\mathsf{supp}'(t) = \mathsf{free}(t)$$

- $\mathsf{free}(a) \overset{\mathsf{def}}{=} \{a\}$
- $\mathsf{free}(t_1 t_2) \overset{\mathsf{def}}{=} \mathsf{free}(t_1) \cup \mathsf{free}(t_1)$
- $\mathsf{free}(\lambda c.t) \overset{\mathsf{def}}{=} \mathsf{free}(t) - \{c\}$

Coming Back to $FTrms$

$$\mathsf{supp}(x) \overset{\mathsf{def}}{=} \{a \mid \mathsf{infinite}\,\{b \mid (a\,b)\bullet x \neq x\}\}$$

Roughly means: the 'free' atoms affected by permutations—this cannot be defined inductively over $\boldsymbol{Ftrms}$.

$$\begin{array}{rcl} t ::= & & \mathsf{er} \\ & \mid & \mathsf{am}(a) \\ & \mid & \mathsf{pr}(t_1, t_2) \\ & \mid & \mathsf{se}(fn) \end{array}$$

We are stuck with supp. . . but this isn't so bad.

Not in the Support

An old friend can be defined in terms of support:

$$a \# x \overset{\mathsf{def}}{=} a \notin \mathsf{supp}(x)$$

Not in the Support

An old friend can be defined in terms of support:

$$a \# x \overset{\text{def}}{=} a \not\in \mathsf{supp}(x)$$

We can (abstractly) prove for every $PType$ (that includes lambda-calculus and $FTrm$s) that:

$$a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$$

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof of an Old Friend

Lemma: $a \mathrel{\#} x \wedge b \mathrel{\#} x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a = b$ clear

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\mathsf{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\mathsf{fin}\{c \mid (b\,c)\bullet x \neq x\}$

$$a \# x \overset{\text{def}}{=} a \notin \mathsf{supp}(x)$$

$$\mathsf{supp}(x) \overset{\text{def}}{=} \{a \mid \mathsf{inf}\{c \mid (a\,c)\bullet x \neq x\}\}$$

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\mathsf{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\mathsf{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2) $\mathsf{fin}(\{c \mid (a\,c)\bullet x \neq x\} \cup \{c \mid (b\,c)\bullet x \neq x\})$ f. (1)

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\mathsf{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\mathsf{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2′) $\mathsf{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\text{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\text{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2′) $\text{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3) $\text{inf}\{c \mid \neg((a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x)\}$ f. (2′)

Given a finite set of atoms,
its 'co-set' must be infinite.

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b){\bullet}x = x$

Proof: case $a \neq b$:

(1) $\mathsf{fin}\{c \mid (a\,c){\bullet}x \neq x\}$ from Ass. +Def. of $\#$

$\mathsf{fin}\{c \mid (b\,c){\bullet}x \neq x\}$

(2′) $\mathsf{fin}\{c \mid (a\,c){\bullet}x \neq x \vee (b\,c){\bullet}x \neq x\}$ f. (1)

(3′) $\mathsf{inf}\{c \mid (a\,c){\bullet}x = x \wedge (b\,c){\bullet}x = x)\}$ f. (2′)

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\, b) \bullet x = x$

Proof: case $a \neq b$:

(1) $\text{fin}\{c \mid (a\, c) \bullet x \neq x\}$ from Ass. +Def. of $\#$

$\text{fin}\{c \mid (b\, c) \bullet x \neq x\}$

(2') $\text{fin}\{c \mid (a\, c) \bullet x \neq x \vee (b\, c) \bullet x \neq x\}$ f. (1)

(3') $\text{inf}\{c \mid (a\, c) \bullet x = x \wedge (b\, c) \bullet x = x)\}$ f. (2')

(4) (i) $(a\, c) \bullet x = x$ (ii) $(b\, c) \bullet x = x$ for a $c \in$ (3')

If a set is infinite, it must contain a few elements.

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\mathrm{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$
$\mathrm{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2') $\mathrm{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3') $\mathrm{inf}\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2')

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3')

(5) $(a\,c)\bullet x = x$ by (4i)

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\text{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\text{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2′) $\text{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3′) $\text{inf}\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2′)

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3′)

(5) $(a\,c)\bullet x = x$ by (4i)

(6) $(b\,c)\bullet(a\,c)\bullet x = (b\,c)\bullet x$ by bij.

bij.: $x = y$ iff $\pi\bullet x = \pi\bullet y$

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\text{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\text{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2') $\text{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3') $\text{inf}\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2')

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3')

(5) $(a\,c)\bullet x = x$ by (4i)

(6') $(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4ii)

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) fin$\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

fin$\{c \mid (b\,c)\bullet x \neq x\}$

(2′) fin$\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3′) inf$\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2′)

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3′)

(5) $(a\,c)\bullet x = x$ by (4i)

(6′) $(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4ii)

(7) $(a\,c)\bullet(b\,c)\bullet(a\,c)\bullet x = (a\,c)\bullet x$ by bij.

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\mathrm{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$

$\mathrm{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2′) $\mathrm{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3′) $\mathrm{inf}\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2′)

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3′)

(5) $(a\,c)\bullet x = x$ by (4i)

(6′) $(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4ii)

(7′) $(a\,c)\bullet(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4i)

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) fin$\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$
fin$\{c \mid (b\,c)\bullet x \neq x\}$

(2′) fin$\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3′) inf$\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2′)

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3′)

(5) $(a\,c)\bullet x = x$ by (4i)

(6′) $(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4ii)

(7′) $(a\,c)\bullet(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4i)

$$(a\,c)(b\,c)(a\,c)\bullet a = b$$
$$(a\,c)(b\,c)(a\,c)\bullet b = a$$
$$(a\,c)(b\,c)(a\,c)\bullet c = c$$

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) fin$\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$
fin$\{c \mid (b\,c)\bullet x \neq x\}$

(2') fin$\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3') inf$\{c \mid$ f. (2')

(4) (i) $(a\,c$ $\in$ (3')

(5) $(a\,c)\bullet x = x$ by (4i)

(6') $(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4ii)

(7') $(a\,c)\bullet(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4i)

(8) $(a\,b)\bullet x = x$ by 3rd. prop.

3rd prop. of permutation types:
$ds(\pi_1, \pi_2) = \varnothing \Rightarrow \pi_1\bullet x = \pi_2\bullet x$

Proof of an Old Friend

Lemma: $a \# x \wedge b \# x \Rightarrow (a\,b)\bullet x = x$

Proof: case $a \neq b$:

(1) $\mathrm{fin}\{c \mid (a\,c)\bullet x \neq x\}$ from Ass. +Def. of $\#$
$\mathrm{fin}\{c \mid (b\,c)\bullet x \neq x\}$

(2′) $\mathrm{fin}\{c \mid (a\,c)\bullet x \neq x \vee (b\,c)\bullet x \neq x\}$ f. (1)

(3′) $\mathrm{inf}\{c \mid (a\,c)\bullet x = x \wedge (b\,c)\bullet x = x)\}$ f. (2′)

(4) (i) $(a\,c)\bullet x = x$ (ii) $(b\,c)\bullet x = x$ for a $c \in$ (3′)

(5) $(a\,c)\bullet x = x$ by (4i)

(6′) $(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4ii)

(7′) $(a\,c)\bullet(b\,c)\bullet(a\,c)\bullet x = x$ by bij.,(4i)

(8) $(a\,b)\bullet x = x$ by 3rd. prop.

Done.

Another Small Proof

Lemma: $\pi \bullet \mathsf{supp}(x) = \mathsf{supp}(\pi \bullet x)$

Another Small Proof

Lemma: $\pi \bullet \mathsf{supp}(x) = \mathsf{supp}(\pi \bullet x)$

Proof:

Another Small Proof

Lemma: $\pi \bullet \mathsf{supp}(x) = \mathsf{supp}(\pi \bullet x)$

Proof:

(1) $\{\pi \bullet a \mid \mathsf{inf}\{b \mid (a\, b) \bullet x \neq x\}\}$ by Def.

$= \{a \mid \mathsf{inf}\{b \mid (a\, b) \bullet \pi \bullet x \neq \pi \bullet x\}\}$

Another Small Proof

Lemma: $\pi \bullet \mathsf{supp}(x) = \mathsf{supp}(\pi \bullet x)$

Proof:

$$
\begin{array}{lcll}
(1) & & \{\pi \bullet a \mid \mathsf{inf}\{b \mid (a\,b) \bullet x \neq x\}\} & \text{by Def.} \\
 & = & \{a \mid \mathsf{inf}\{b \mid (a\,b) \bullet \pi \bullet x \neq \pi \bullet x\}\} & \\
(2) & = & \{a \mid \mathsf{inf}\{b \mid \pi^{-1} \bullet (a\,b) \bullet \pi \bullet x \neq x\}\} &
\end{array}
$$

Another Small Proof

Lemma: $\pi\bullet\mathsf{supp}(x) = \mathsf{supp}(\pi\bullet x)$

Proof:

$$
\begin{array}{lcl}
(1) & & \{\pi\bullet a \mid \mathsf{inf}\{b \mid (a\,b)\bullet x \neq x\}\} \qquad \text{by Def.} \\
 & = & \{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet\pi\bullet x \neq \pi\bullet x\}\} \\
(2) & = & \{a \mid \mathsf{inf}\{b \mid \pi^{-1}\bullet(a\,b)\bullet\pi\bullet x \neq x\}\} \\
(3) & = & \{a \mid \mathsf{inf}\{b \mid (\pi^{-1}\bullet a\ \ \pi^{-1}\bullet b)\bullet x \neq x\}\}
\end{array}
$$

Another Small Proof

Lemma: $\pi{\bullet}\mathsf{supp}(x) = \mathsf{supp}(\pi{\bullet}x)$

Proof:

$$\begin{array}{lcll}
(1) & & \{\pi{\bullet}a \mid \mathsf{inf}\{b \mid (a\,b){\bullet}x \neq x\}\} & \text{by Def.} \\
 & = & \{a \mid \mathsf{inf}\{b \mid (a\,b){\bullet}\pi{\bullet}x \neq \pi{\bullet}x\}\} & \\
(2) & = & \{a \mid \mathsf{inf}\{b \mid \pi^{-1}{\bullet}(a\,b){\bullet}\pi{\bullet}x \neq x\}\} & \\
(3) & = & \{a \mid \mathsf{inf}\{b \mid (\pi^{-1}{\bullet}a\ \ \pi^{-1}{\bullet}b){\bullet}x \neq x\}\} & \\
(4) & = & \{\pi{\bullet}a \mid \mathsf{inf}\{\pi{\bullet}b \mid (a\,b){\bullet}x \neq x\}\} &
\end{array}$$

Another Small Proof

Lemma: $\pi \bullet \mathsf{supp}(x) = \mathsf{supp}(\pi \bullet x)$

Proof:

(1) $\{\pi \bullet a \mid \mathsf{inf}\{b \mid (a\,b) \bullet x \neq x\}\}$ by Def.

$= \{a \mid \mathsf{inf}\{b \mid (a\,b) \bullet \pi \bullet x \neq \pi \bullet x\}\}$

(2) $= \{a \mid \mathsf{inf}\{b \mid \pi^{-1} \bullet (a\,b) \bullet \pi \bullet x \neq x\}\}$

(3) $= \{a \mid \mathsf{inf}\{b \mid (\pi^{-1} \bullet a \;\; \pi^{-1} \bullet b) \bullet x \neq x\}\}$

(4) $= \{\pi \bullet a \mid \mathsf{inf}\{\pi \bullet b \mid (a\,b) \bullet x \neq x\}\}$

(5) the set $\{b \mid (a\,b) \bullet x \neq x\}$ is infinite, (1)+(4)
whenever $\{\pi \bullet b \mid (a\,b) \bullet x \neq x\}$ is and v-v.

Another Small Proof

Lemma: $\pi\bullet\mathsf{supp}(x) = \mathsf{supp}(\pi\bullet x)$

Proof:

(1) $\{\pi\bullet a \mid \mathsf{inf}\{b \mid (a\,b)\bullet x \neq x\}\}$ by Def.

$= \{a \mid \mathsf{inf}\{b \mid (a\,b)\bullet\pi\bullet x \neq \pi\bullet x\}\}$

(2) $= \{a \mid \mathsf{inf}\{b \mid \pi^{-1}\bullet(a\,b)\bullet\pi\bullet x \neq x\}\}$

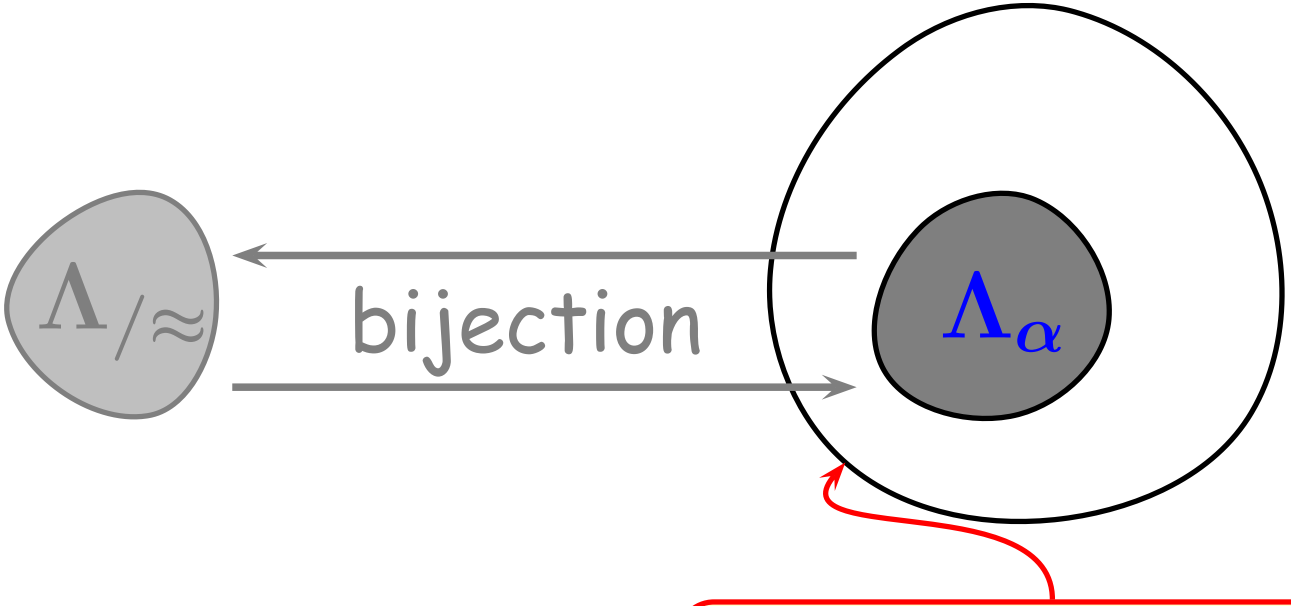
(3) $= \{a \mid \mathsf{inf}\{b \mid (\pi^{-1}\bullet a\;\;\pi^{-1}\bullet b)\bullet x \neq x\}\}$

(4) $= \{\pi\bullet a \mid \mathsf{inf}\{\pi\bullet b \mid (a\,b)\bullet x \neq x\}\}$

(5) the set $\{b \mid (a\,b)\bullet x \neq x\}$ is infinite, (1)+(4)
whenever $\{\pi\bullet b \mid (a\,b)\bullet x \neq x\}$ is and v-v.

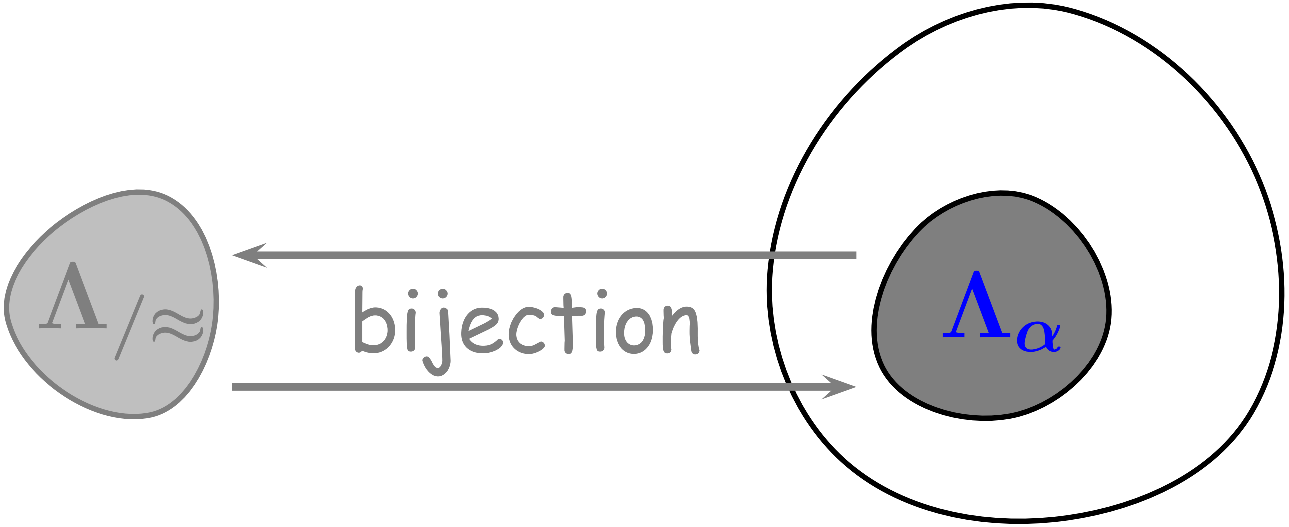
Done.

What About Small-Set?



$$\begin{array}{rcl} t ::= & & \mathsf{er} \\ & | & \mathsf{am}(a) \\ & | & \mathsf{pr}(t_1, t_2) \\ & | & \mathsf{se}(fn) \end{array}$$

What About Small-Set?



For Λ_α, we are only interested in some very specific functions, namely

$$
\begin{aligned}
[a].t \overset{\text{def}}{=} \text{se}\ (\lambda b.\ & \text{if } a = b \\
& \text{then } t \\
& \text{else if } b \# t \text{ then } (b\,a) \bullet t \text{ else er})
\end{aligned}
$$

Function $[a].t$ '=' $[\lambda a.t]_\alpha$

$$[a].t \overset{\text{def}}{=} \text{se } (\lambda b.\ \begin{array}{l} \text{if } a = b \\ \text{then } t \\ \text{else if } b \mathrel{\#} t \text{ then } (b\,a) \bullet t \text{ else er}) \end{array}$$

Function $[a].t$ ‘$=$’ $[\lambda a.t]_\alpha$

$$[a].t \overset{\text{def}}{=} \text{se}\ (\lambda b.\ \begin{array}{l} \text{if } a = b \\ \text{then } t \\ \text{else if } b \# t \text{ then } (b\,a)\bullet t \text{ else er}) \end{array}$$

This is supposed to stand for the α-equivalence class of $\lambda a.t$.

Function $[a].t$ ‘$=$’ $[\lambda a.t]_\alpha$

```
$[a].\mathsf{pr}(a,c) \overset{\mathsf{def}}{=}$
        $\mathsf{se}\ (\lambda b.\ \mathsf{if}\ a = b$
                $\mathsf{then}\ \mathsf{pr}(a,c)$
                $\mathsf{else}\ \mathsf{if}\ b \mathbin{\#} \mathsf{pr}(a,c)$
                    $\mathsf{then}\ (b\,a) \bullet \mathsf{pr}(a,c)\ \mathsf{else}\ \mathsf{er})$
```

Let’s check this for $[a].\mathsf{pr}(a,c)$:

Function $[a].t$ ‘$=$’ $[\lambda a.t]_\alpha$

$[a].\mathsf{pr}(a,c) \overset{\mathsf{def}}{=}$

```
        se (λb. if $a = b$
                  then $\mathsf{pr}(a,c)$
                  else if $b \# \mathsf{pr}(a,c)$
                       then $(b\,a) \bullet \mathsf{pr}(a,c)$ else er)
```

Let’s check this for $[a].\mathsf{pr}(a,c)$:

$[a].\mathsf{pr}(a,c)$ ‘applied to’ a ‘gives’ $\mathsf{pr}(a,c)$

Function $[a].t$ ‘$=$’ $[\boldsymbol{\lambda} a.t]_\alpha$

```
$[a].\mathsf{pr}(a, c) \overset{\mathsf{def}}{=}$
        se $(\lambda b.$ if $a = b$
                then pr$(a, c)$
                else if $b \# \mathsf{pr}(a, c)$
                    then $(b\, a) \bullet \mathsf{pr}(a, c)$ else er$)$
```

Let’s check this for $[a].\mathsf{pr}(a, c)$:

$[a].\mathsf{pr}(a, c)$ ‘applied to’ a ‘gives’ $\mathsf{pr}(a, c)$

$[a].\mathsf{pr}(a, c)$ ‘applied to’ b ‘gives’ $\mathsf{pr}(b, c)$

Function $[a].t$ ‘$=$’ $[\lambda a.t]_\alpha$

```
$[a].\mathsf{pr}(a,c) \overset{\mathsf{def}}{=}$
        $\mathsf{se}\ (\lambda b.\ \mathsf{if}\ a = b$
                $\mathsf{then}\ \mathsf{pr}(a,c)$
                $\mathsf{else}\ \mathsf{if}\ b \# \mathsf{pr}(a,c)$
                    $\mathsf{then}\ (b\,a)\bullet\mathsf{pr}(a,c)\ \mathsf{else}\ \mathsf{er})$
```

Let’s check this for $[a].\mathsf{pr}(a,c)$:

$[a].\mathsf{pr}(a,c)$ ‘applied to’ a ‘gives’ $\mathsf{pr}(a,c)$

$[a].\mathsf{pr}(a,c)$ ‘applied to’ b ‘gives’ $\mathsf{pr}(b,c)$

$[a].\mathsf{pr}(a,c)$ ‘applied to’ c ‘gives’ er

Function $[a].t$ ‘$=$’ $[\lambda a.t]_\alpha$

$$
\begin{array}{l}
[a].\mathsf{pr}(a,c) \overset{\mathsf{def}}{=} \\
\qquad\qquad \mathsf{se}\ (\lambda b.\ \mathsf{if}\ a = b \\
\qquad\qquad\qquad\qquad \mathsf{then}\ \mathsf{pr}(a,c) \\
\qquad\qquad\qquad\qquad \mathsf{else}\ \mathsf{if}\ b \,\#\, \mathsf{pr}(a,c) \\
\qquad\qquad\qquad\qquad\qquad \mathsf{then}\ (b\,a) \bullet \mathsf{pr}(a,c)\ \mathsf{else}\ \mathsf{er})
\end{array}
$$

Let's check this for $[a].\mathsf{pr}(a,c)$:

$[a].\mathsf{pr}(a,c)$ 'applied to' a 'gives' $\mathsf{pr}(a,c)$

$[a].\mathsf{pr}(a,c)$ 'applied to' b 'gives' $\mathsf{pr}(b,c)$

$[a].\mathsf{pr}(a,c)$ 'applied to' c 'gives' er

$[a].\mathsf{pr}(a,c)$ 'applied to' d 'gives' $\mathsf{pr}(d,c)$

$\vdots$

Function $[a].t$ ‘$=$’ $[\lambda a.t]_\alpha$

```
$[a].\mathsf{pr}(a,c) \overset{\mathsf{def}}{=}$
        se $(\lambda b.$ if $a = b$
                then $\mathsf{pr}(a,c)$
                else if $b \# \mathsf{pr}(a,c)$
                    then $(b\,a)\bullet\mathsf{pr}(a,c)$ else er$)$
```

Let’s check this for $[a].\mathsf{pr}(a,c)$:

$[a].\mathsf{pr}(a,c)$ ‘applied to’ a ‘gives’ $\mathsf{pr}(a,c)$ ‘$\lambda a.(a\,c)$’

$[a].\mathsf{pr}(a,c)$ ‘applied to’ b ‘gives’ $\mathsf{pr}(b,c)$ ‘$\lambda b.(b\,c)$’

$[a].\mathsf{pr}(a,c)$ ‘applied to’ c ‘gives’ er

$[a].\mathsf{pr}(a,c)$ ‘applied to’ d ‘gives’ $\mathsf{pr}(d,c)$ ‘$\lambda d.(d\,c)$’

$\vdots$

Function $[a].t$ ‘=’ $[\lambda a.t]_{\alpha}$

$[a].\mathrm{pr}(a, c) \overset{\mathrm{def}}{=}$

```
se ($\lambda b.$ if $a = b$
        then $\mathrm{pr}(a, c)$
        else if $b \# \mathrm{pr}(a, c)$
            then $(b\,a) \bullet \mathrm{pr}(a, c)$ else er)
```

Let's check this for $[a].\mathrm{pr}(a, c)$:

$[a].\mathrm{pr}(a, c)$ ‘applied to’ a ‘gives’ $\mathrm{pr}(a, c)$

$[a].\mathrm{pr}(a, c)$ ‘applied to’ b ‘gives’ $\mathrm{pr}(b, c)$

$[a].\mathrm{pr}(a, c)$ ‘applied to’ c ‘gives’ er

$[a].\mathrm{pr}(a, c)$ ‘applied to’ d ‘gives’ $\mathrm{pr}(d, c)$

$\vdots$

$[\lambda a.(a\,c)]_{tl\alpha}$:

‘$\lambda a.(a\,c)$’

‘$\lambda b.(b\,c)$’

‘$\lambda d.(d\,c)$’

$\vdots$

Properties of $[a].t$

- $\pi \bullet ([a].t) = [\pi \bullet a].(\pi \bullet t)$

Should be familiar from Monday:

$$\pi \bullet \boldsymbol{\lambda} a.t \overset{\text{def}}{=} \boldsymbol{\lambda}(\pi \bullet a).(\pi \bullet t)$$

(a simple calculation for $[a].t$)

Properties of $[a].t$

- $\pi\bullet([a].t) = [\pi\bullet a].(\pi\bullet t)$
- $t_1 = t_2 \Leftrightarrow [a].t_1 = [a].t_2$
- $a \neq b \Rightarrow (t_1 = (a\,b)\bullet t_2 \;\wedge\; a \# t_2$
  $\Leftrightarrow [a].t_1 = [b].t_2)$

Should also be familiar from Monday:

$$\frac{t_1 \approx t_2}{\lambda a.t_1 \approx \lambda a.t_2} \qquad \frac{a \neq b \quad t_1 \approx (a\,b)\bullet t_2 \quad a \# t_2}{\lambda a.t_1 \approx \lambda b.t_2}$$

Properties of $[a].t$

- $\pi \bullet ([a].t) = [\pi \bullet a].(\pi \bullet t)$
- $t_1 = t_2 \Leftrightarrow [a].t_1 = [a].t_2$
- $a \neq b \Rightarrow (t_1 = (a\,b) \bullet t_2 \ \wedge \ a \# t_2$
  $\Leftrightarrow [a].t_1 = [b].t_2)$

These properties (plus the $Ptype$ properties and one further restriction on t), will give:

- $a \# [a].t$
- $a \neq b \wedge a \# t \Leftrightarrow a \# [b].t$
- $\mathsf{supp}([a].t) = \mathsf{supp}(t) - \{a\}$

Properties of $[a].t$

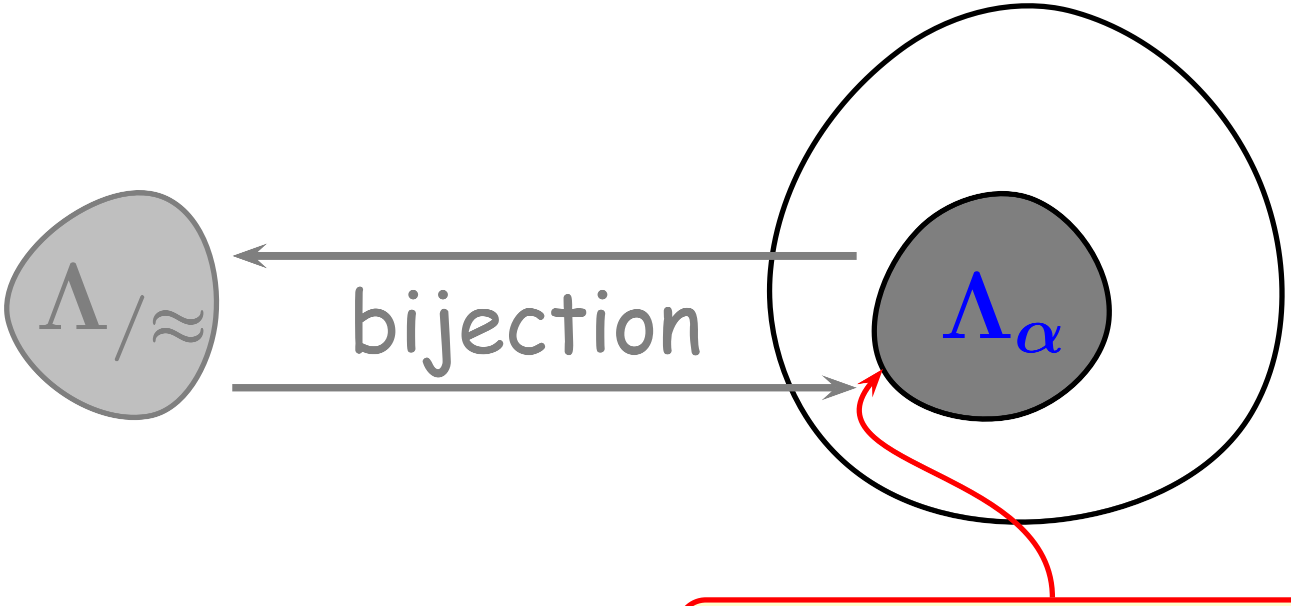
- $\pi \bullet ([a].t) = [\pi \bullet a].(\pi \bullet t)$
- $t_1 = t_2 \Leftrightarrow [a].t_1 = [a].t_2$
- $a \neq b$ $a \# t_2$ $t_1 = [b].t_2)$

These prop properties and one fur will give:

So $[a].t$ behaves very much like what we would expect from a lambda-abstraction.

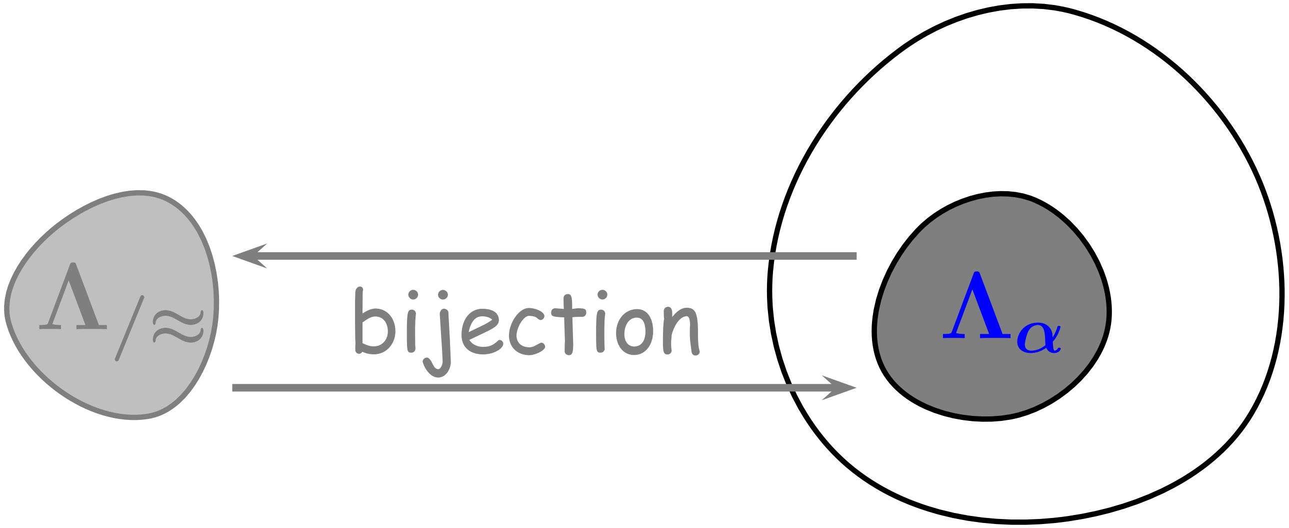
- $a \# [a].t$
- $a \neq b \wedge a \# t \Leftrightarrow a \# [b].t$
- $\text{supp}([a].t) = \text{supp}(t) - \{a\}$

Definition of Small-Set



$$
\begin{array}{lcl}
t ::= & & \mathsf{am}(a) \\
 & | & \mathsf{pr}(t_1, t_2) \\
 & | & [a].t
\end{array}
$$

Definition of Small-Set



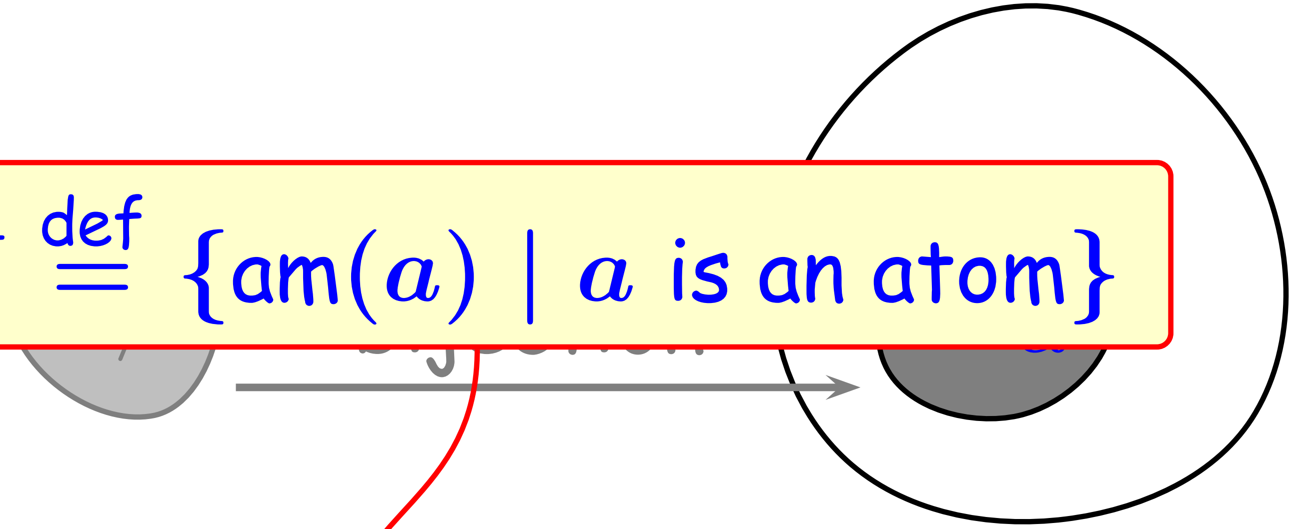
$$F(X) \overset{\text{def}}{=} AM \cup PR(X) \cup AS(X)$$

$$\Lambda_\alpha \overset{\text{def}}{=} \mathsf{lfp}(F) = \bigcup_n F_n$$

where $F_0 \overset{\text{def}}{=} F(\varnothing)$

$F_{n+1} \overset{\text{def}}{=} F(F_n)$

Definition of Small-Set



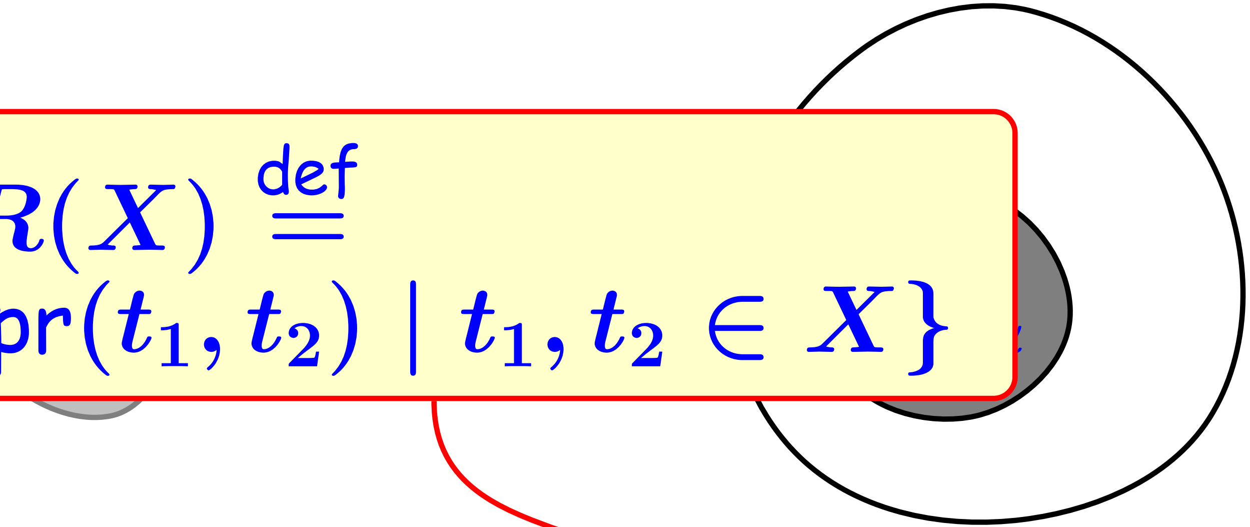
$$AM \overset{\text{def}}{=} \{\mathsf{am}(a) \mid a \text{ is an atom}\}$$

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{X}) &\overset{\text{def}}{=} \boldsymbol{AM} \cup \boldsymbol{PR}(\boldsymbol{X}) \cup \boldsymbol{AS}(\boldsymbol{X}) \\ \Lambda_\alpha &\overset{\text{def}}{=} \mathsf{lfp}(\boldsymbol{F}) = \textstyle\bigcup_n \boldsymbol{F}_n \end{aligned}$$

where $\boldsymbol{F}_0 \overset{\text{def}}{=} \boldsymbol{F}(\varnothing)$

$\boldsymbol{F}_{n+1} \overset{\text{def}}{=} \boldsymbol{F}(\boldsymbol{F}_n)$

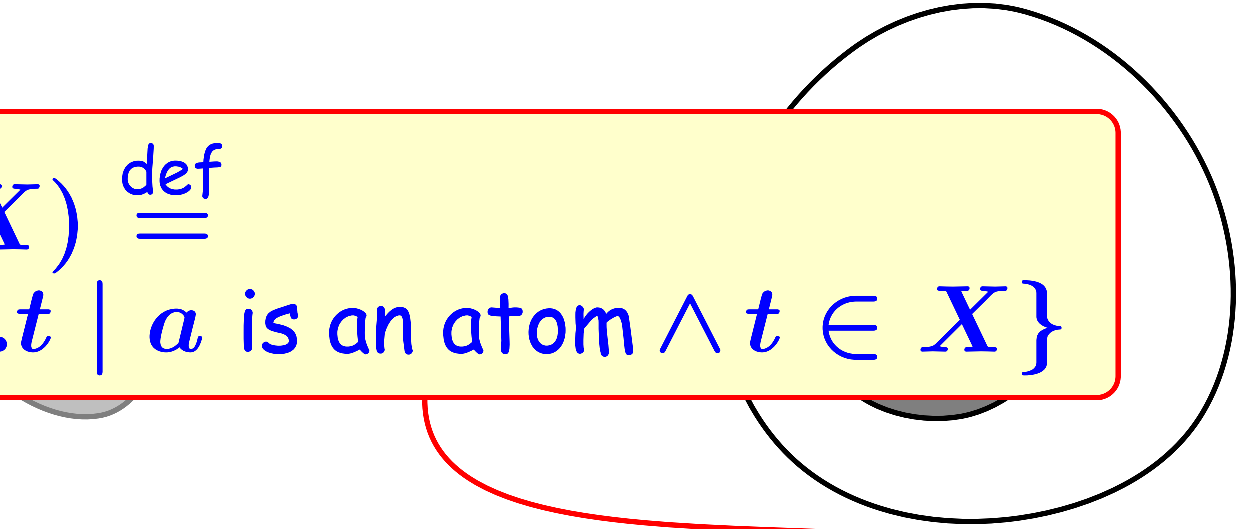
Definition of Small-Set



$$PR(X) \overset{\text{def}}{=} \{\mathsf{pr}(t_1, t_2) \mid t_1, t_2 \in X\}$$

$$
\begin{aligned}
F(X) &\overset{\text{def}}{=} AM \cup PR(X) \cup AS(X)\\
\Lambda_\alpha &\overset{\text{def}}{=} \mathsf{lfp}(F) \;=\; \bigcup\nolimits_n F_n\\
&\qquad \text{where } F_0 \overset{\text{def}}{=} F(\varnothing)\\
&\qquad \phantom{\text{where }} F_{n+1} \overset{\text{def}}{=} F(F_n)
\end{aligned}
$$

Definition of Small-Set



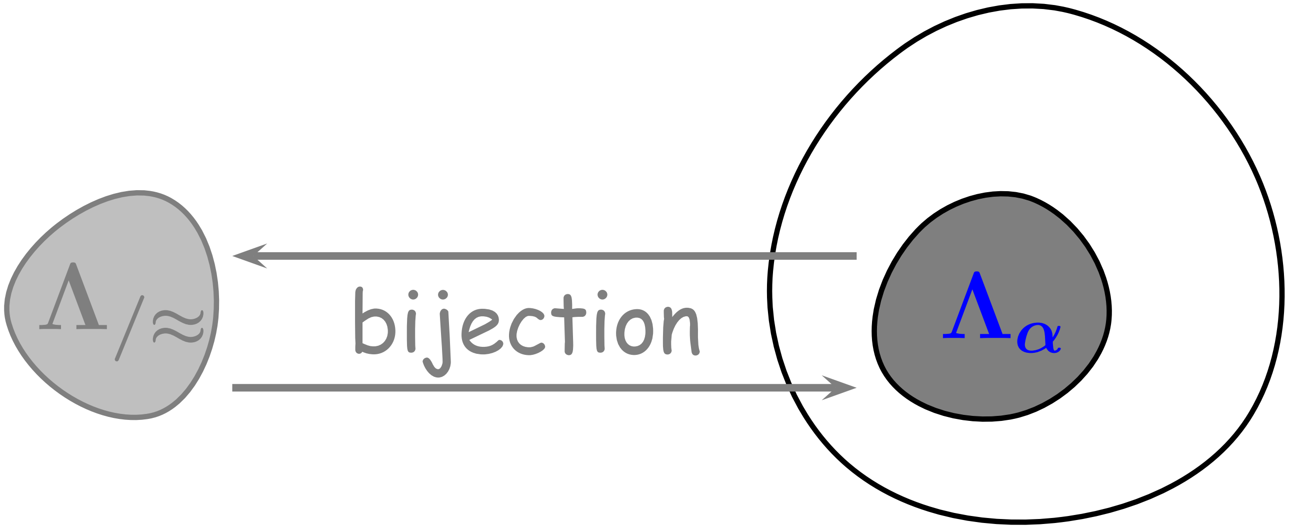
$$\boldsymbol{AS(X)} \overset{\text{def}}{=} \{[a].t \mid a \text{ is an atom} \wedge t \in \boldsymbol{X}\}$$

$$\begin{aligned}
\boldsymbol{F(X)} &\overset{\text{def}}{=} \boldsymbol{AM} \cup \boldsymbol{PR(X)} \cup \boldsymbol{AS(X)} \\
\Lambda_\alpha &\overset{\text{def}}{=} \text{lfp}(\boldsymbol{F}) = \textstyle\bigcup_n \boldsymbol{F}_n
\end{aligned}$$

where $\boldsymbol{F}_0 \overset{\text{def}}{=} \boldsymbol{F}(\varnothing)$

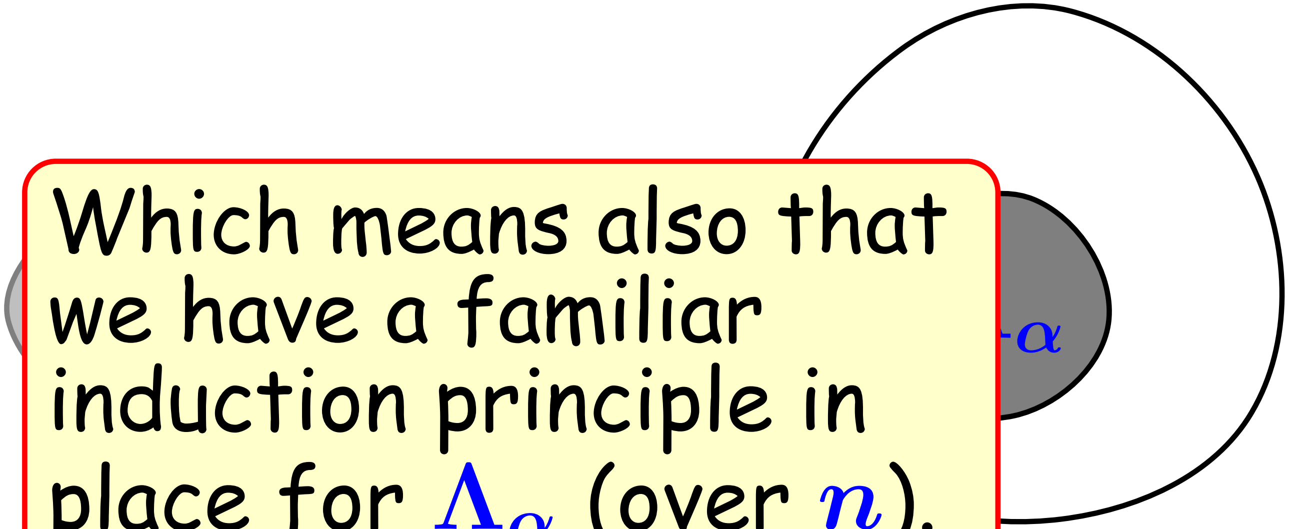
$\boldsymbol{F}_{n+1} \overset{\text{def}}{=} \boldsymbol{F}(\boldsymbol{F}_n)$

Definition of Small-Set



$$
\begin{aligned}
F(X) &\overset{\text{def}}{=} AM \cup PR(X) \cup AS(X)\\
\Lambda_\alpha &\overset{\text{def}}{=} \text{lfp}(F) = \textstyle\bigcup_n F_n\\
&\quad \text{where } F_0 \overset{\text{def}}{=} F(\varnothing)\\
&\qquad\quad\; F_{n+1} \overset{\text{def}}{=} F(F_n)
\end{aligned}
$$

Definition of Small-Set



Which means also that we have a familiar induction principle in place for Λ_α (over n).

$$\begin{aligned}
F(X) &\overset{\text{def}}{=} AM \cup PR(X) \cup AS(X) \\
\Lambda_\alpha &\overset{\text{def}}{=} \mathsf{lfp}(F) = \bigcup_n F_n \\
&\quad \text{where} \quad F_0 \overset{\text{def}}{=} F(\varnothing) \\
&\quad \phantom{\text{where} \quad} F_{n+1} \overset{\text{def}}{=} F(F_n)
\end{aligned}$$

Finite Support

$$\mathsf{fsupp}(x) \overset{\mathsf{def}}{=} \mathsf{finite}(\mathsf{supp}(x))$$

While an $\boldsymbol{Ftrm}$ is not necessarily finitely supported, every element in $\boldsymbol{\Lambda_\alpha}$ is.

- $\mathsf{supp}(\mathsf{am}(a)) = \{a\}$
- $\mathsf{supp}(\mathsf{pr}(t_1, t_2)) = \mathsf{supp}(t_1) \cup \mathsf{supp}(t_2)$
- $\mathsf{supp}([a].t) = \mathsf{supp}(t) - \{a\}$

Finite Support

$$\mathsf{fsupp}(x) \overset{\mathsf{def}}{=} \mathsf{finite}(\mathsf{supp}(x))$$

While an $\boldsymbol{Ftrm}$ is not necessarily finitely supported, every element in $\boldsymbol{\Lambda_\alpha}$ is.

- $\mathsf{supp}(\mathsf{am}(a)) = \{a\}$
- $\mathsf{supp}(\mathsf{pr}(t_1, t_2)) = \mathsf{supp}(t_1) \cup \mathsf{supp}(t_2)$
- $\mathsf{supp}([a].t) = \mathsf{supp}(t) - \{a\}$

Whenever an x is finitely supported, then

$$(\exists a : \boldsymbol{Atom})\ a \# x \text{ !!}$$

Finite Support

fsup $\overset{\text{def}}{}$ $(x))$

If a $PType$ is finitely supported, then we call it an $FSType$.

While an Ft y finitely supported, e s.

- supp(am$(a)) = \{a\}$
- supp(pr$(t_1, t_2)) =$ supp$(t_1) \cup$ supp(t_2)
- supp$([a].t) =$ supp$(t) - \{a\}$

Whenever an x is finitely supported, then

$$(\exists a : Atom)\ a \# x$$!!

Bijection

In order to show that $\Lambda_{/\approx}$ and Λ_α are bijective we define a function q from Λ to Λ_α:

$$\begin{aligned} q(a) &\stackrel{\text{def}}{=} \mathsf{am}(a) \\ q(t_1\, t_2) &\stackrel{\text{def}}{=} \mathsf{pr}(q(t_1), q(t_2)) \\ q(\lambda a.t) &\stackrel{\text{def}}{=} [a].q(t) \end{aligned}$$

with the property

$$t_1 \approx t_2 \;\Leftrightarrow\; q(t_1) = q(t_2)$$

Bijection

Aside: This is as close to the 'bijection' as you possibly want, but you **can** get closer: you can 'lift' q to $\Lambda_{/\approx}$. A theorem prover doesn't let you easily choose one element from a set; with all elements it is no problem. So q' can be defined as

$$q'(X) \overset{\text{def}}{=} \{q(t) \mid t \in X\}$$

If q behaves well with respect to the α-equivalence class, then we defined a singleton set. Stripping of the set-brackets gives you a function from $\Lambda_{/\approx}$ to Λ_{α}.

Λ_α is an $FSType$

i.e., a finitely supported $PType$. It inherits the following properties from $Ftrm$

- $\pi \bullet ([a].t) = [\pi \bullet a].(\pi \bullet t)$
- $t_1 = t_2 \Leftrightarrow [a].t_1 = [a].t_2$
- $a \neq b \Rightarrow (t_1 = (a\,b) \bullet t_2 \;\wedge\; a \,\#\, t_2 \Leftrightarrow [a].t_1 = [b].t_2)$

Λ_α is an $FSType$

i.e., a finitely supported $PType$. It inherits the following properties from $Ftrm$

- $\pi\bullet([a].t) = [\pi\bullet a].(\pi\bullet t)$
- $t_1 = t_2 \Leftrightarrow [a].t_1 = [a].t_2$

To remind you, the important properties we have already shown are:

- $a \# x \wedge b \# x \Rightarrow (a\, b)\bullet x = x$
- $a \# x \Leftrightarrow \pi\bullet a \# \pi\bullet x$

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Proof:

(1) $(\exists c) c \# (a, b, t, [a].t)$ "finitely supported"

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Proof:

(1) $(\exists c) c \# (a, b, t, [a].t)$ "finitely supported"

(2) $(b\, c) \bullet t = t$ from (1) + ass.

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Proof:

(1) $(\exists c) c \# (a, b, t, [a].t)$ "finitely supported"

(2) $(b\, c) \bullet t = t$ from (1) + ass.

(3) $(b\, c) \bullet c \# (b\, c) \bullet [a].t$ from $c \# [a].t$

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Proof:

(1) $(\exists c) c \# (a, b, t, [a].t)$ "finitely supported"

(2) $(b\,c) \bullet t = t$ from (1) + ass.

(3) $b \# [a].((b\,c) \bullet t)$ from $c \# [a].t$

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Proof:

(1) $(\exists c) c \# (a, b, t, [a].t)$ “finitely supported”

(2) $(b\,c) \bullet t = t$ from (1) + ass.

(3) $b \# [a].((b\,c) \bullet t)$ from $c \# [a].t$

(4) $b \# [a].t$ (2)+(3)

Freshness 1

Lemma: $a \neq b \wedge b \# t \Rightarrow b \# [a].t$

Proof:

(1) $(\exists c) c \# (a, b, t, [a].t)$ "finitely supported"

(2) $(b\, c) \bullet t = t$ from (1) + ass.

(3) $b \# [a].((b\, c) \bullet t)$ from $c \# [a].t$

(4) $b \# [a].t$ (2)+(3)

Done.