Lecture 5: nominal ADTs

Algebraic Datatypes

Nominal sets support a syntax-free notion of α-equivalence.

Q: What can we do with it?

Nominal Algebraic Datatypes

Nominal sets support a syntax-free notion of α-equivalence.

Q: What can we do with it?

A: Inductively defined data types in $\mathbf{Nom}$ that combine the usual operations

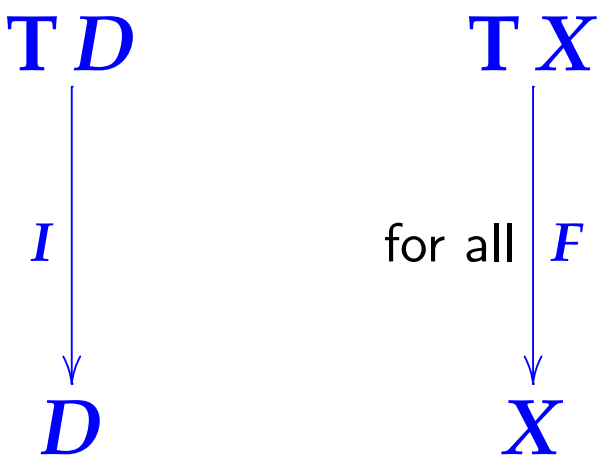
- disjoint union $X_1 + \cdots + X_n$
  (for data with different constructor forms)
- cartesian product $X_1 \times \cdots \times X_n$
  (for constructor of several arguments)

with

- name-abstraction $[\mathbb{A}]X$
  (for constructors that involve binding).

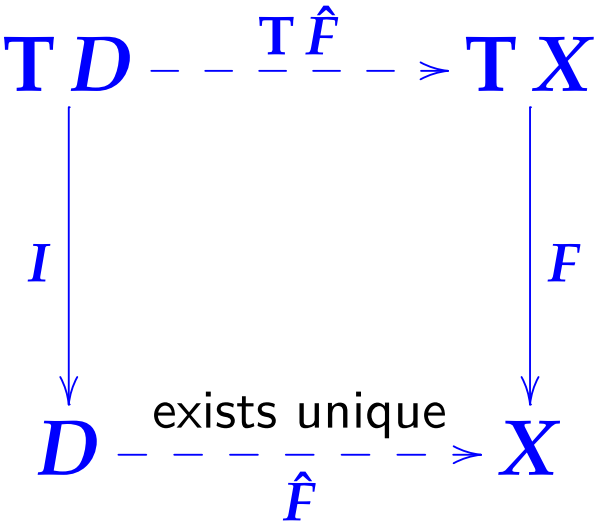
Initial algebras

- $[\mathbb{A}](-)$ has excellent exactness properties. It can be combined with $\times$, $+$ (and $X \to_{\mathbf{fs}} (-)$) to give functors $\mathbf{T} : \mathbf{Nom} \to \mathbf{Nom}$ that have initial algebras $I : \mathbf{T}\, D \to D$



Initial algebras

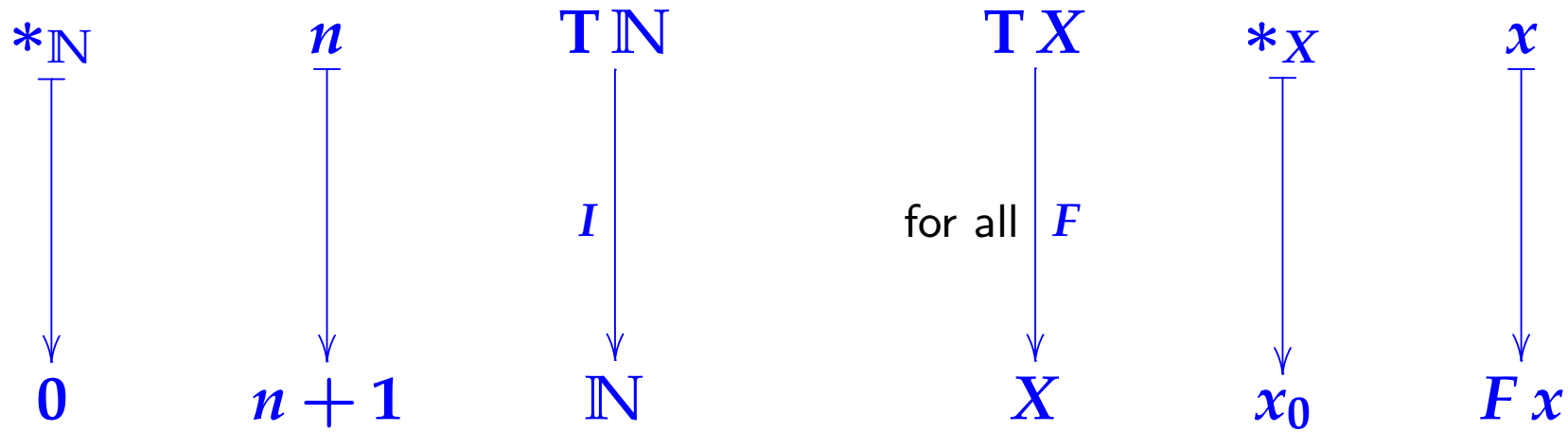
- $[\mathbb{A}](-)$ has excellent exactness properties. It can be combined with $\times$, $+$ (and $X \to_{\mathbf{fs}} (-)$) to give functors $\mathbf{T} : \mathbf{Nom} \to \mathbf{Nom}$ that have initial algebras $I : \mathbf{T}\,D \to D$



E.g. $\mathbb{N}$ as an initial algebra

for $T(-) = 1 + (-) : \mathbf{Set} \to \mathbf{Set}$.

Concretely, take $T(X) = \{*_X\} \cup X$ for some $*_X \notin X$.



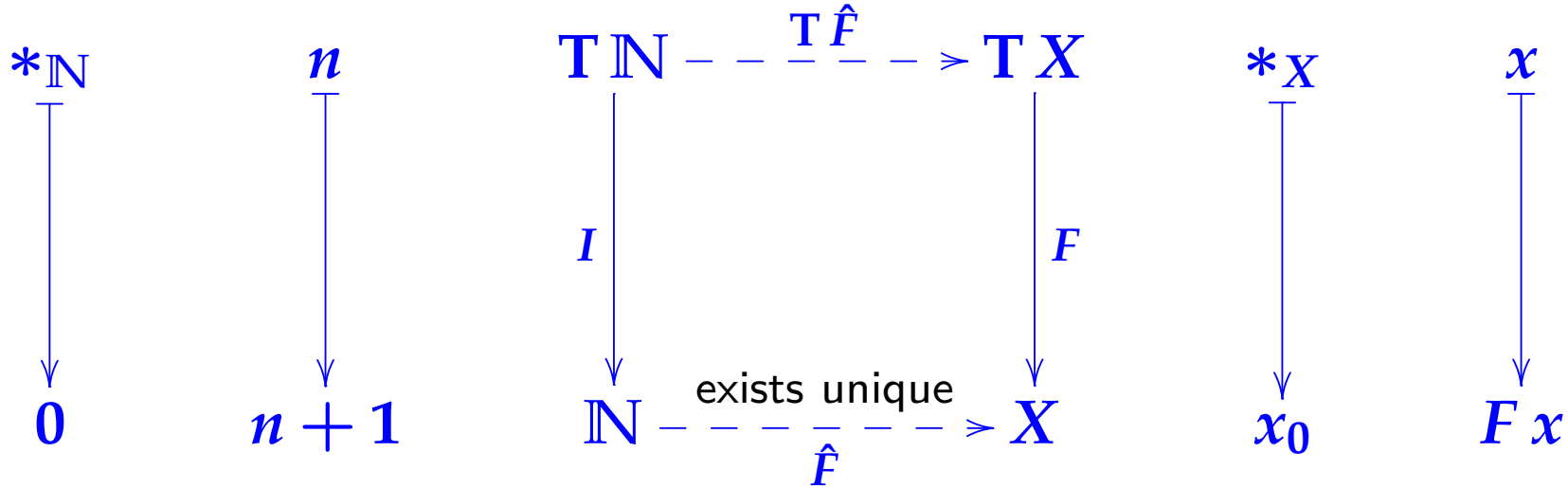
E.g. $\mathbb{N}$ as an initial algebra

for $T(-) = 1 + (-) : \mathbf{Set} \to \mathbf{Set}$.

Concretely, take $T(X) = \{*_X\} \cup X$ for some $*_X \notin X$.

initial algebra

some algebra



iterative definition: $\begin{cases} \hat{F}\,0 = x_0 \\ \hat{F}(n+1) = F(\hat{F}\,n) \end{cases}$

(special case of primitive recursion)

Initial algebras

- $[\mathbb{A}](-)$ has excellent exactness properties. It can be combined with $\times$, $+$ (and $X \to_{\mathbf{fs}} (-)$) to give functors $\mathbf{T} : \mathbf{Nom} \to \mathbf{Nom}$ that have initial algebras $I : \mathbf{T}\, D \to D$
- For a wide class of such functors (nominal algebraic functors) the initial algebra D coincides with ASTs/α-equivalence.
  E.g. Λ is the initial algebra for

$$\mathbf{T}(-) \triangleq \mathbb{A} + (- \times -) + [\mathbb{A}](-)$$

Nominal algebraic signatures

- Sorts $\mathtt{S} ::= \mathtt{N}$ name-sort (here just one, for simplicity)
  $\mid \mathtt{D}$ data-sorts
  $\mid \mathtt{1}$ unit
  $\mid \mathtt{S},\mathtt{S}$ pairs
  $\mid \mathtt{N}.\mathtt{S}$ name-binding
- Typed operations $\mathtt{op} : \mathtt{S} \to \mathtt{D}$

Signature Σ is specified by the stuff in red.

Nominal algebraic signatures

Example: λ-calculus

name-sort `Var` for variables, data-sort `Term` for terms, and operations

$$\mathtt{V} : \mathtt{Var} \longrightarrow \mathtt{Term}$$

$$\mathtt{A} : \mathtt{Term}, \mathtt{Term} \longrightarrow \mathtt{Term}$$

$$\mathtt{L} : \mathtt{Var}.\mathtt{Term} \longrightarrow \mathtt{Term}$$

Nominal algebraic signatures

Example: λ-calculus $+\texttt{let rec}\ f(x) = e\ \texttt{in}\ e'$

name-sort $\texttt{Var}$ for variables, data-sort $\texttt{Term}$ for terms, and operations

$$\texttt{V} : \texttt{Var} \longrightarrow \texttt{Term}$$

$$\texttt{A} : \texttt{Term}, \texttt{Term} \longrightarrow \texttt{Term}$$

$$\texttt{L} : \texttt{Var}.\texttt{Term} \longrightarrow \texttt{Term}$$

$$\texttt{R} : \texttt{Var}.((\texttt{Var}.\texttt{Term}), \texttt{Term}) \longrightarrow \texttt{Term}$$

Nominal algebraic signatures

Example: λ-calculus + $\texttt{let rec}\ f(x) = e\ \texttt{in}\ e'$

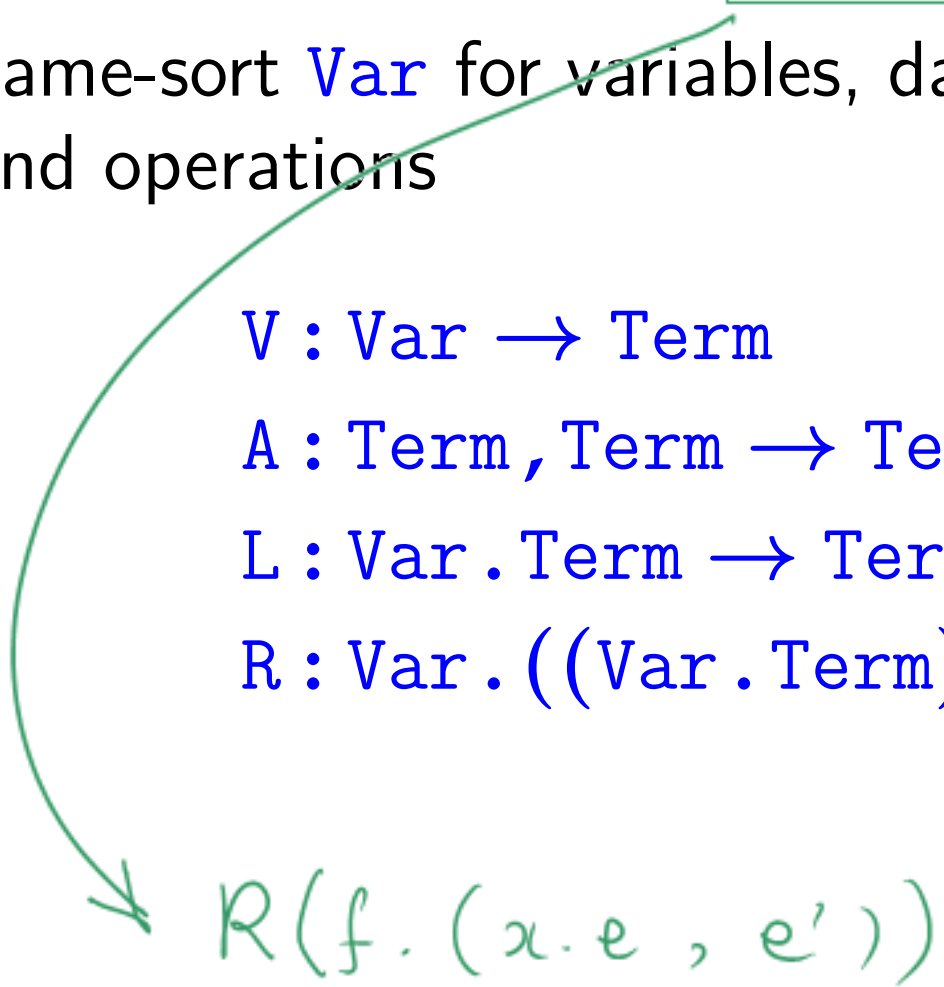
name-sort Var for variables, data-sort Term for terms, and operations

$$\texttt{V} : \texttt{Var} \longrightarrow \texttt{Term}$$

$$\texttt{A} : \texttt{Term}, \texttt{Term} \longrightarrow \texttt{Term}$$

$$\texttt{L} : \texttt{Var}.\texttt{Term} \longrightarrow \texttt{Term}$$

$$\texttt{R} : \texttt{Var}.((\texttt{Var}.\texttt{Term}), \texttt{Term}) \longrightarrow \texttt{Term}$$



Nominal algebraic signatures

Closely related notions:

- *binding signatures* of Fiore, Plotkin & Turi (LICS 1999)
- *nominal algebras* of Honsell, Miculan & Scagnetto (ICALP 2001)

N.B. all these notions of signature restrict attention to iterated, but *unary* name-binding—there are other kinds of lexically scoped binder (e.g. see Pottier's Cαml language, or Urban's Nominal 2 package for Isabell/HOL.)

$\Sigma(\mathtt{S})$ = raw terms over Σ of sort $\mathtt{S}$

$$\frac{a \in \mathbb{A}}{a \in \Sigma(\mathtt{N})} \qquad \frac{t \in \Sigma(\mathtt{S}) \qquad \mathtt{op} : \mathtt{S} \to \mathtt{D}}{\mathtt{op}\, t \in \Sigma(\mathtt{D})} \qquad \frac{}{() \in \Sigma(1)}$$

$$\frac{t_1 \in \Sigma(\mathtt{S}_1) \qquad t_2 \in \Sigma(\mathtt{S}_2)}{t_1 , t_2 \in \Sigma(\mathtt{S}_1 , \mathtt{S}_2)} \qquad \frac{a \in \mathbb{A} \qquad t \in \Sigma(\mathtt{S})}{a \,.\, t \in \Sigma(\mathtt{N} \,.\, \mathtt{S})}$$

Each $\Sigma(\mathtt{S})$ is a nominal set once equipped with the obvious $\mathbf{Perm}\,\mathbb{A}$-action—any finite set of atoms containing all those occurring in t supports $t \in \Sigma(\mathtt{S})$.

Alpha-equivalence

$$=_\alpha \subseteq \Sigma(\mathrm{S}) \times \Sigma(\mathrm{S})$$

$$\frac{a \in \mathbb{A}}{a =_\alpha a} \qquad \frac{t =_\alpha t'}{\mathrm{op}\, t =_\alpha \mathrm{op}\, t'} \qquad \frac{}{() =_\alpha ()}$$

$$\frac{t_1 =_\alpha t_1' \qquad t_2 =_\alpha t_2'}{t_1, t_2 =_\alpha t_1', t_2'}$$

$$\frac{(a_1\ a) \cdot t_1 =_\alpha (a_2\ a) \cdot t_2 \qquad a \# (a_1, t_1, a_2, t_2)}{a_1 . t_1 =_\alpha a_2 . t_2}$$

Alpha-equivalence

$$=_\alpha \subseteq \Sigma(\mathbb{S}) \times \Sigma(\mathbb{S})$$

Fact: $=_\alpha$ is equivariant ($t_1 =_\alpha t_2 \Rightarrow \pi \cdot t_1 =_\alpha \pi \cdot t_2$) and each quotient

$$\Sigma_\alpha(\mathbb{S}) \triangleq \{[t]_\alpha \mid t \in \Sigma(\mathbb{S})\}$$

is a nominal set with

$$\begin{array}{rcl} \pi \cdot [t]_\alpha & = & [\pi \cdot t]_\alpha \\ supp\,[t]_\alpha & = & fn\,t \end{array}$$

where

$$\begin{array}{rcl} fn(a\,.\,t) & = & fn\,t - \{a\} \\ fn(t_1\,,\,t_2) & = & fn\,t_1 \cup fn\,t_2 \end{array}$$

etc.

Theorem. Given a nominal algebraic signature Σ (for simplicity, assume Σ has a single data-sort $\mathtt{D}$ as well as a single name-sort $\mathtt{N}$)
$\Sigma_\alpha(\mathtt{D})$ is an initial algebra for the associated functor $\mathbf{T}_\Sigma : \mathbf{Nom} \to \mathbf{Nom}$.

(NSB p139.)

Theorem. Given a nominal algebraic signature Σ (for simplicity, assume Σ has a single data-sort $\mathtt{D}$ as well as a single name-sort $\mathtt{N}$)
$\Sigma_\alpha(\mathtt{D})$ is an initial algebra for the associated functor $\mathbf{T}_\Sigma : \mathbf{Nom} \to \mathbf{Nom}$.

$$\mathbf{T}_\Sigma(-) = [\![\mathtt{S}_1]\!](-) + \cdots + [\![\mathtt{S}_n]\!](-)$$

where Σ has operations $\mathtt{op}_i : \mathtt{S}_i \to D$ $(i = 1..n)$

and $[\![\mathtt{S}]\!](-) : \mathbf{Nom} \to \mathbf{Nom}$ is defined by:

$$\begin{aligned}
[\![\mathtt{N}]\!](-) &= \mathbb{A} \\
[\![\mathtt{D}]\!](-) &= (-) \\
[\![1]\!](-) &= \mathbf{1} \\
[\![\mathtt{S}_1, \mathtt{S}_2]\!](-) &= [\![\mathtt{S}_1]\!](-) \times [\![\mathtt{S}_2]\!](-) \\
[\![\mathtt{N}.\mathtt{S}]\!](-) &= [\mathbb{A}]([\![\mathtt{S}]\!](-))
\end{aligned}$$

Theorem. Given a nominal algebraic signature $\mathbf{\Sigma}$ (for simplicity, assume $\mathbf{\Sigma}$ has a single data-sort $\mathtt{D}$ as well as a single name-sort $\mathtt{N}$)
$\mathbf{\Sigma}_\alpha(\mathtt{D})$ is an initial algebra for the associated functor $\mathbf{T}_\Sigma : \mathbf{Nom} \to \mathbf{Nom}$.

E.g. for the λ-calculus signature with operations

$\mathtt{V} : \mathtt{Var} \longrightarrow \mathtt{Term}$

$\mathtt{A} : \mathtt{Term}, \mathtt{Term} \longrightarrow \mathtt{Term}$

$\mathtt{L} : \mathtt{Var}.\mathtt{Term} \longrightarrow \mathtt{Term}$

we have

$$\mathbf{T}_\Sigma(-) = \mathbb{A} + (- \times -) + [\mathbb{A}](-)$$

Theorem. Given a nominal algebraic signature Σ (for simplicity, assume Σ has a single data-sort $\mathtt{D}$ as well as a single name-sort $\mathtt{N}$)
$\Sigma_\alpha(\mathtt{D})$ is an initial algebra for the associated enriched functor $\mathbf{T}_\Sigma : \mathbf{Nom} \to \mathbf{Nom}$.

$\mathbf{T}_\Sigma$ not only acts on equivariant (=emptily supported) functions, but also on finitely supported functions:

$$\begin{array}{rcl} (X \to_{\mathbf{fs}} Y) & \to & (\mathbf{T}_\Sigma X \to_{\mathbf{fs}} \mathbf{T}_\Sigma Y) \\ F & \mapsto & \mathbf{T}_\Sigma F \end{array}$$

E.g. the enriched functor $[\mathbb{A}](-) : \mathbf{Nom} \to \mathbf{Nom}$ sends $f \in X \to_{\mathbf{fs}} Y$ to $[\mathbb{A}]f \in [\mathbb{A}]X \to_{\mathbf{fs}} [\mathbb{A}]Y$ where

$$[\mathbb{A}]f\,(\langle a\rangle x) = \langle a\rangle(f\,x) \quad \text{if } a \# f$$

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\begin{cases} f_1 \in \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 \in X \times X \to_{\mathbf{fs}} X \\ f_3 \in [\mathbb{A}]X \to_{\mathbf{fs}} X \end{cases}$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \begin{cases} \hat{f}\, a = f_1\, a \\ \hat{f}\,(e_1\, e_2) = f_2(\hat{f}\, e_1, \hat{f}\, e_2) \\ \hat{f}(\lambda a.e) = f_3(\langle a \rangle \hat{f}\, e) \quad \text{if } a \,\#\, (f_1, f_2, f_3) \end{cases}$$

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\begin{cases} f_1 & \in & \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 & \in & X \times X \to_{\mathbf{fs}} X \\ f_3 & \in & [\mathbb{A}]X \to_{\mathbf{fs}} X \end{cases}$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \begin{cases} \hat{f}\, a = f_1\, a \\ \hat{f}\,(e_1\, e_2) = f_2(\hat{f}\, e_1, \hat{f}\, e_2) \\ \hat{f}(\lambda a.e) = f_3(\langle a \rangle \hat{f}\, e) \quad \text{if } a \# (f_1, f_2, f_3) \end{cases}$$

Theorem 2^+. $f \in (\mathbb{A} \times X) \to_{\mathbf{fs}} Y$ gives rise to a unique element of $\overline{f} \in ([\mathbb{A}]X) \to_{\mathbf{fs}} Y$ satisfying

$$\overline{f}(\langle a \rangle x) = f(a, x) \quad \text{if } a \# f$$

iff $(\forall a \in \mathbb{A})\ a \# f \ \Rightarrow\ (\forall x \in X)\ a \# f(a, x)$
(iff $(\exists a \in \mathbb{A})\ a \# f \ \wedge\ (\forall x \in X)\ a \# f(a, x)$).

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\begin{cases} f_1 \in \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 \in X \times X \to_{\mathbf{fs}} X \\ f_3 \in \mathbb{A} \times X \to_{\mathbf{fs}} X \end{cases}$ s.t.

$$(\forall a)\ a \# (f_1, f_2, f_3) \Rightarrow (\forall x)\ a \# f_3(a, x) \qquad \text{(FCB)}$$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \begin{cases} \hat{f}\ a = f_1\ a \\ \hat{f}\ (e_1\ e_2) = f_2(\hat{f}\ e_1, \hat{f}\ e_2) \\ \hat{f}(\lambda a.e) = f_3(a, \hat{f}\ e) \quad \text{if } a \# (f_1, f_2, f_3) \end{cases}$$

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\begin{cases} f_1 \in \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 \in X \times X \to_{\mathbf{fs}} X \\ f_3 \in \mathbb{A} \times X \to_{\mathbf{fs}} X \end{cases}$ s.t.

$$(\forall a)\ a \# (f_1, f_2, f_3) \Rightarrow (\forall x)\ a \# f_3(a, x) \qquad \text{(FCB)}$$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \begin{cases} \hat{f}\ a = f_1\ a \\ \hat{f}\ (e_1\ e_2) = f_2(\hat{f}\ e_1, \hat{f}\ e_2) \\ \hat{f}(\lambda a.e) = f_3(a, \hat{f}\ e) \quad \text{if } a \# (f_1, f_2, f_3) \end{cases}$$

E.g. capture-avoiding substitution $(-)[e'/a'] : \Lambda \to \Lambda$ is the $\hat{f}$ for

$$\begin{array}{rcl} f_1\ a & \triangleq & \textbf{if } a = a' \textbf{ then } e' \textbf{ else } a \\ f_2(e_1, e_2) & \triangleq & e_1\ e_2 \\ f_3(a, e) & \triangleq & \lambda a.e \end{array}$$

for which (FCB) holds, since $a \# \lambda a.e$

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\begin{cases} f_1 & \in & \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 & \in & X \times X \to_{\mathbf{fs}} X \\ f_3 & \in & \mathbb{A} \times X \to_{\mathbf{fs}} X \end{cases}$ s.t.

$$(\forall a)\ a \# (f_1, f_2, f_3) \implies (\forall x)\ a \# f_3(a, x) \qquad \text{(FCB)}$$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \begin{cases} \hat{f}\, a = f_1\, a \\ \hat{f}\,(e_1\, e_2) = f_2(\hat{f}\, e_1, \hat{f}\, e_2) \\ \hat{f}(\lambda a.e) = f_3(a, \hat{f}\, e) \quad \text{if } a \# (f_1, f_2, f_3) \end{cases}$$

E.g. size function $\Lambda \to \mathbb{N}$ is the $\hat{f}$ for

$$\begin{array}{rcl} f_1\, a & \triangleq & 0 \\ f_2(n_1, n_2) & \triangleq & n_1 + n_2 \\ f_3(a, n) & \triangleq & n + 1 \end{array}$$

for which (FCB) holds, since $a \# (n + 1)$

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\begin{cases} f_1 \in \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 \in X \times X \to_{\mathbf{fs}} X \\ f_3 \in \mathbb{A} \times X \to_{\mathbf{fs}} X \end{cases}$ s.t.

$$(\forall a)\ a \# (f_1, f_2, f_3) \Rightarrow (\forall x)\ a \# f_3(a, x) \qquad \text{(FCB)}$$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \begin{cases} \hat{f}\, a = f_1\, a \\ \hat{f}\,(e_1\, e_2) = f_2(\hat{f}\, e_1, \hat{f}\, e_2) \\ \hat{f}(\lambda a.e) = f_3(a, \hat{f}\, e) \quad \text{if } a \# (f_1, f_2, f_3) \end{cases}$$

Non-example: trying to list the bound variables of a λ-term

$$\begin{array}{rcl} f_1\, a & \triangleq & \mathbf{nil} \\ f_2(\ell_1, \ell_2) & \triangleq & \ell_1 @ \ell_2 \\ f_3(a, \ell) & \triangleq & a :: \ell \end{array}$$

for which (FCB) does not hold, since $a \in supp(a :: \ell)$.

α-Structural recursion

For λ-terms:

Theorem.
Given any $X \in \mathbf{Nom}$ and $\left\{ \begin{array}{lcl} f_1 & \in & \mathbb{A} \to_{\mathbf{fs}} X \\ f_2 & \in & X \times X \to_{\mathbf{fs}} X \\ f_3 & \in & \mathbb{A} \times X \to_{\mathbf{fs}} X \end{array} \right.$ s.t.

$$(\forall a)\ a \# (f_1, f_2, f_3) \ \Rightarrow\ (\forall x)\ a \# f_3(a, x) \qquad \text{(FCB)}$$

$$\exists!\ \hat{f} \in \Lambda \to_{\mathbf{fs}} X \text{ s.t. } \left\{ \begin{array}{rl} \hat{f}\, a = f_1\, a & \\ \hat{f}\,(e_1\, e_2) = f_2(\hat{f}\, e_1, \hat{f}\, e_2) & \\ \hat{f}(\lambda a.e) = f_3(a, \hat{f}\, e) & \text{if } a \# (f_1, f_2, f_3) \end{array} \right.$$

Similar results hold for any nominal algebraic signature—see J ACM 53(2006)459–506.

Implemented in Urban & Berghofer's Nominal package for Isabelle/HOL (classical higher-order logic).

Seems to capture informal usage well, but (FCB) can be tricky...

Counting bound variables

For each $e \in \Lambda$, $\boxed{\mathbf{cbv}\, e \triangleq f\, e\, \rho_0 \in \mathbb{N}}$

where we want $f \in \Lambda \to_{\mathbf{fs}} X$ with
$X = (\mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to_{\mathbf{fs}} \mathbb{N}$ to satisfy

$$\begin{aligned} f\, a\, \rho &= \rho\, a \\ f\, (e_1\, e_2)\, \rho &= (f\, e_1 \rho) + (f\, e_2\, \rho) \\ f\, (\lambda a.e)\, \rho &= f\, e\, (\rho[a \mapsto \mathbf{1}]) \end{aligned}$$

and where $\rho_0 \in \mathbb{A} \to_{\mathbf{fs}} \mathbb{N}$ is $\lambda(a \in \mathbb{A}) \to \mathbf{0}$.

Counting bound variables

For each $e \in \Lambda$, $\boxed{\mathbf{cbv}\, e \triangleq f\, e\, \rho_0 \in \mathbb{N}}$

where we want $f \in \Lambda \to_{\mathbf{fs}} X$ with
$X = (\mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to_{\mathbf{fs}} \mathbb{N}$ to satisfy

$$\begin{aligned} f\, a\, \rho &= \rho\, a \\ f\, (e_1\, e_2)\, \rho &= (f\, e_1 \rho) + (f\, e_2\, \rho) \\ f\, (\lambda a.e)\, \rho &= f\, e\, (\rho[a \mapsto \mathbf{1}]) \end{aligned}$$

and where $\rho_0 \in \mathbb{A} \to_{\mathbf{fs}} \mathbb{N}$ is $\lambda(a \in \mathbb{A}) \to \mathbf{0}$.

Looks like we should take
$f_3(a, x) = \lambda(\rho \in \mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to x(\rho[a \mapsto \mathbf{1}])$,
but this does not satisfy (FCB). Solution: take X to be a certain nominal subset of $(\mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to_{\mathbf{fs}} \mathbb{N}$. (See NSB p145.)

Counting bound variables

For each $e \in \Lambda$, $\quad \boxed{\mathbf{cbv}\, e \triangleq f\, e\, \rho_0 \in \mathbb{N}}$

where we want $f \in \Lambda \to_{\mathbf{fs}} X$ with $X = (\mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to_{\mathbf{fs}} \mathbb{N}$ to satisfy

$$\begin{aligned} f\, a\, \rho &= \rho\, a \\ f\, (e_1\, e_2)\, \rho &= (f\, e_1 \rho) + (f\, e_2\, \rho) \\ f\, (\lambda a.e)\, \rho &= f\, e\, (\rho[a \mapsto \mathbf{1}]) \end{aligned}$$

$\{x \mid (\forall a)(\forall \rho)(\forall n)\ x(\rho[a \mapsto n]) = x\rho\}$

and where $\rho_0 \in \mathbb{A} \to_{\mathbf{fs}} \mathbb{N}$ is $\lambda(a \in \mathbb{A}) \to \mathbf{0}$.

Looks like we should take
$f_3(a, x) = \lambda(\rho \in \mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to x(\rho[a \mapsto \mathbf{1}])$,
but this does not satisfy (FCB). Solution: take X to be a certain nominal subset of $(\mathbb{A} \to_{\mathbf{fs}} \mathbb{N}) \to_{\mathbf{fs}} \mathbb{N}$. (See NSB p145.)