Quiz

Assuming that a and b are distinct variables, is it possible to find λ-terms M_1 to M_7 that make the following pairs α-equivalent?

- $\lambda a.\lambda b.(M_1\, b)$ and $\lambda b.\lambda a.(a\, M_1)$
- $\lambda a.\lambda b.(M_2\, b)$ and $\lambda b.\lambda a.(a\, M_3)$
- $\lambda a.\lambda b.(b\, M_4)$ and $\lambda b.\lambda a.(a\, M_5)$
- $\lambda a.\lambda b.(b\, M_6)$ and $\lambda a.\lambda a.(a\, M_7)$

If there is one solution for a pair, can you describe all its solutions?

Nominal Unification

Christian Urban
Andrew Pitts
Jamie Gabbay

University of Cambridge

Nominal Unification

Why?

- First-order unification is simple, but cannot be used for terms involving binders.
- Higher-order unification is (more) complicated — e.g. Huet's algorithms or L_λ by Miller — and not satisfactory from a pragmatic point of view (not always decidable, not always MGUs or applies only to a restricted class of terms).

. . . and Substitution

Higher-order: capture-avoiding substitution
But often one wants to use possibly-capturing substitution (or context-substitution)

for example:

$$\frac{\mathtt{app}(\mathtt{fn}\,\boldsymbol{a}.\boldsymbol{Y},\boldsymbol{X}) \Downarrow \boldsymbol{V}}{\mathtt{let}\,\boldsymbol{a} = \boldsymbol{X}\,\mathtt{in}\,\boldsymbol{Y} \Downarrow \boldsymbol{V}}$$

...and Substitution

Higher-order: capture-avoiding substitution
But often one wants to use possibly-capturing substitution (or context-substitution)

for example:

$$\frac{\mathtt{app}(\mathtt{fn}\, \boldsymbol{a}.\, \boldsymbol{a},\ 1\ ) \Downarrow\ 1}{\mathtt{let}\ \boldsymbol{a} = 1\ \mathtt{in}\ \boldsymbol{a} \Downarrow\ 1}$$

- $\mathtt{let}\ \boldsymbol{a} = 1\ \mathtt{in}\ \boldsymbol{a} \Downarrow 1 \quad [\boldsymbol{Y} := \boldsymbol{a}; \boldsymbol{X}, \boldsymbol{V} := 1]$

...and Substitution

Higher-order: capture-avoiding substitution
But often one wants to use possibly-capturing substitution (or context-substitution)

for example:

$$\frac{\mathtt{app}(\mathtt{fn}\; a.\, b,\; 1\;) \Downarrow 1}{\mathtt{let}\; b = 1 \;\mathtt{in}\; b \Downarrow 1}$$

error!

- $\mathtt{let}\; a = 1 \;\mathtt{in}\; a \Downarrow 1 \qquad [Y := a; X, V := 1]$
- $\mathtt{let}\; b = 1 \;\mathtt{in}\; b \Downarrow 1 \qquad [Y := b; X, V := 1]$

…and Substitution

Higher-order: capture-avoiding substitution
But often one wants to use possibly-capturing substitution (or context-substitution)

for example:

$$\frac{\mathtt{app}(\mathtt{fn}\,\boldsymbol{\lambda a.Fa})\,\boldsymbol{X} \;\Downarrow\; \boldsymbol{V}}{\mathtt{let}\,\boldsymbol{X}(\boldsymbol{\lambda a.Fa}) \;\Downarrow\; \boldsymbol{V}}$$

...and Substitution

Higher-order: capture-avoiding substitution
But often one wants to use possibly-capturing substitution (or context-substitution)

for example:

$$\frac{\mathrm{app}(\mathtt{fn}\ F)\ X\ \Downarrow\ V}{\mathtt{let}\ X\ F\ \Downarrow\ V}$$

- $\mathtt{let}\ 1\ \lambda a.a \Downarrow 1$ or $\mathtt{let}\ 1\ \lambda b.b \Downarrow 1$

...and Substitution

Higher-order: capture-avoiding substitution
But often one wants to use possibly-capturing substitution (or context-substitution)

for example:

$$\frac{\texttt{app}(\texttt{fn}\ \boldsymbol{F})\ \boldsymbol{X}\ \Downarrow\ \boldsymbol{V}}{\texttt{let}\ \boldsymbol{X}\ \boldsymbol{F}\ \Downarrow\ \boldsymbol{V}}$$

- $\texttt{let}\ 1\ \boldsymbol{\lambda a.a} \Downarrow 1$ or $\texttt{let}\ 1\ \boldsymbol{\lambda b.b} \Downarrow 1$

Does it have to be so? No!

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$\texttt{fn}\ \boldsymbol{a.b} \qquad\qquad \texttt{fn}\ \boldsymbol{c.b}$$

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$\begin{aligned}[b := a]\,\mathrm{fn}\; a.b \\ = \mathrm{fn}\; a.a\end{aligned} \qquad \begin{aligned}[b := a]\,\mathrm{fn}\; c.b \\ = \mathrm{fn}\; c.a\end{aligned}$$

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$[b := a]\,\mathrm{fn}\, a.b = \mathrm{fn}\, a.a \qquad [b := a]\,\mathrm{fn}\, c.b = \mathrm{fn}\, c.a$$

<u>Traditional Solution:</u> replace $[b := a]t$ by a more complicated, 'capture-avoiding' form of substitution.

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$(b\,a)\cdot \mathtt{fn}\; a.b = \mathtt{fn}\; b.a \qquad\qquad (b\,a)\cdot \mathtt{fn}\; c.b = \mathtt{fn}\; c.a$$

Nice Alternative: use a less complicated operation for renaming

$(b\,a)\cdot t \stackrel{\text{def}}{=}$ swap **all** occurrences of b and a in t

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$(b\,a)\cdot\,\texttt{fn}\ a.b = \texttt{fn}\ b.a \qquad\qquad (b\,a)\cdot\,\texttt{fn}\ c.b = \texttt{fn}\ c.a$$

Nice Alternative: use a less complicated operation for renaming

$(b\,a)\cdot t \stackrel{\text{def}}{=}$ swap **all** occurrences of b and a in t

be they free, bound or binding

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$(b\,a)\cdot \mathrm{fn}\, a.b = \mathrm{fn}\, b.a \qquad (b\,a)\cdot \mathrm{fn}\, c.b = \mathrm{fn}\, c.a$$

<u>Nice Alternative:</u> use a less complicated operation for renaming

$(b\,a)\cdot t \stackrel{\text{def}}{=}$ swap **all** occurrences of b and a in t

Unlike for $[b:=a](-)$, for $(b\,a)\cdot(-)$ we do have if $t =_\alpha t'$ then $(b\,a)\cdot t =_\alpha (b\,a)\cdot t'$.

Swappings

Problem: substitution does not respect α-equivalence, e.g.

$$(b\,a)\cdot\,\mathrm{fn}\,a.b = \mathrm{fn}\,b.a \qquad\qquad (b\,a)\cdot\,\mathrm{fn}\,c.b = \mathrm{fn}\,c.a$$

Nice Alternative: use a less complicated

operati

$($

Unlike

have if $t =_\alpha t'$ then $(b\,a)\cdot t =_\alpha (b\,a)\cdot t'$.

Preview:

In the next few slides we shall extend 'swappings' to 'lists of swappings'

$$(a_1\,b_1)\ldots(a_n\,b_n),$$

also called **permutations**.

Terms

- $\langle\rangle$ Units
- $\langle t, t'\rangle$ Pairs
- $F\, t$ Funct.

Terms

- $\langle\rangle$ Units
- a Atoms
- $\langle t, t' \rangle$ Pairs
- $F\,t$ Funct.

Terms

- $\langle\rangle$ Units
- $\langle t, t' \rangle$ Pairs
- $F\,t$ Funct.
- a Atoms
- $a.t$ Abstractions

Terms

- $\langle\rangle$ Units
- a Atoms
- $\langle t, t' \rangle$ Pairs
- $a.t$ Abstractions
- $F\,t$ Funct.

$\ulcorner \lambda a.a \urcorner \longmapsto \mathtt{fn}\, a.a$

constructions like $\mathtt{fn}\, X.X$ are not allowed

Terms

- $\langle\rangle$ Units
- a Atoms
- $\langle t, t'\rangle$ Pairs
- $a.t$ Abstractions
- $F\,t$ Funct.
- $\pi{\cdot}X$ Suspensions

Terms

- $\langle\rangle$ Units
- a Atoms
- $\langle t, t' \rangle$ Pairs
- $a.t$ Abstractions
- $F\,t$ Funct.
- $\pi{\cdot}X$ Suspensions

π is an explicit permutation, which is a list of swappings $(a_1\, b_1) \ldots (a_n\, b_n)$, waiting to be applied to the term that is substituted for X

X is a meta-level variable, standing for an unknown term

Permutations

a permutation applied to a term:

- $$[]\cdot a \stackrel{\text{def}}{=} a$$
- $$(b\,c)::\pi\cdot a \stackrel{\text{def}}{=} \begin{cases} c & \text{if } \pi\cdot a = b \\ b & \text{if } \pi\cdot a = c \\ \pi\cdot a & \text{otherwise} \end{cases}$$

Permutations

a permutation applied to a term:

- $[]{\cdot}a \stackrel{\text{def}}{=} a$
- $(b\,c) :: \pi\,{\cdot}a \stackrel{\text{def}}{=} \begin{cases} c & \text{if } \pi{\cdot}a = b \\ b & \text{if } \pi{\cdot}a = c \\ \pi{\cdot}a & \text{otherwise} \end{cases}$
- $\pi{\cdot}a.t \stackrel{\text{def}}{=} \pi{\cdot}a.\pi{\cdot}t$

Permutations

a permutation applied to a term:

- $$[]\cdot a \stackrel{\text{def}}{=} a$$
- $$(b\,c)::\pi\cdot a \stackrel{\text{def}}{=} \begin{cases} c & \text{if } \pi\cdot a = b \\ b & \text{if } \pi\cdot a = c \\ \pi\cdot a & \text{otherwise} \end{cases}$$
- $$\pi\cdot a.t \stackrel{\text{def}}{=} \pi\cdot a.\pi\cdot t$$
- $$\pi\cdot\pi'\cdot X \stackrel{\text{def}}{=} (\pi @ \pi')\cdot X$$

Permutations

a permutation applied to a term:

- $[]\cdot a \stackrel{\text{def}}{=} a$
- $(b\,c)::\pi\cdot a \stackrel{\text{def}}{=} \begin{cases} c & \text{if } \pi\cdot a = b \\ b & \text{if } \pi\cdot a = c \\ \pi\cdot a & \text{otherwise} \end{cases}$
- $\pi\cdot a.t \stackrel{\text{def}}{=} \pi\cdot a.\pi\cdot t$
- $\pi\cdot\pi'\cdot X \stackrel{\text{def}}{=} (\pi @ \pi')\cdot X$

Permutations on atoms are bijections!

$$\pi\cdot a = b \quad \text{iff} \quad a = (\pi^{-1})\cdot b$$

Freshness Relation

We will identify

$$\mathtt{fn}\, a.X \;\approx\; \mathtt{fn}\, b.(a\, b)\cdot X$$

provided that 'b is fresh for X — ($b \# X$)',
i.e., does not occur freely in any ground term
that might be substituted for X.

Freshness Relation

We will identify

$$\mathrm{fn}\, a.X \;\approx\; \mathrm{fn}\, b.(a\, b)\cdot X$$

provided that 'b is fresh for X — ($b \# X$)'

i.e., does not occur

that might be subs

explicit permutation — waits to be applied to the term that is substituted for X

Freshness Relation

We will identify

$$\mathtt{fn}\, a.X \;\approx\; \mathtt{fn}\, b.(a\, b)\cdot X$$

provided that 'b is fresh for X — ($b \# X$)', i.e., does not occur freely in any ground term that might be substituted for X.

Freshness Relation

We will identify

$$\mathtt{fn}\, a.X \;\approx\; \mathtt{fn}\, b.(a\, b){\cdot}X$$

provided that 'b is fresh for X — $(b \# X)$',
i.e., does not occur freely in any ground term
that might be substituted for X.

If we know more about X, e.g., if we knew
that $a \# X$ and $b \# X$, then we can replace
$(a\, b){\cdot}X$ by X.

Freshness Assumptions

Our equality is **not** just

$$t \approx t' \qquad \alpha\text{-equivalence}$$

Freshness Assumptions

but judgements

$$\nabla \vdash t \approx t' \qquad \alpha\text{-equivalence}$$

where

$$\nabla = \{a_1 \# X_1, \ldots, a_n \# X_n\}$$

is a finite set of freshness assumptions.

Freshness Assumptions

but judgements

$$\nabla \vdash t \approx t' \qquad \alpha\text{-equivalence}$$

where

$$\nabla = \{a_1 \# X_1, \ldots, a_n \# X_n\}$$

is a finite set of freshness assumptions.

$$\{a \# X, b \# X\} \vdash \mathtt{fn}\, a.X \approx \mathtt{fn}\, b.X$$

Freshness Assumptions

but judgements

$$\nabla \vdash t \approx t' \qquad \alpha\text{-equivalence}$$

$$\nabla \vdash a \# t \qquad \text{freshness}$$

where

$$\nabla = \{a_1 \# X_1, \ldots, a_n \# X_n\}$$

is a finite set of freshness assumptions.

$$\{a \# X, b \# X\} \vdash \mathtt{fn}\, a.X \approx \mathtt{fn}\, b.X$$

Rules for Equivalence

Excerpt
(i.e. only the interesting rules)

Rules for Equivalence

$$\frac{\nabla \vdash t \approx t'}{\nabla \vdash a.t \approx a.t'}$$

$$\frac{a \neq b \qquad \nabla \vdash t \approx (a\,b)\cdot t' \qquad \nabla \vdash a \,\#\, t'}{\nabla \vdash a.t \approx b.t'}$$

Rules for Equivalence

$$\frac{(a \# X) \in \nabla \quad \text{for all } a \text{ with } \pi{\cdot}a \neq \pi'{\cdot}a}{\nabla \vdash \pi{\cdot}X \approx \pi'{\cdot}X}$$

Rules for Equivalence

$$\frac{(a \,\#\, X) \in \nabla \quad \text{for all } a \text{ with } \pi{\cdot}a \neq \pi'{\cdot}a}{\nabla \vdash \pi{\cdot}X \approx \pi'{\cdot}X}$$

for example

$$\{a \,\#\, X, b \,\#\, X\} \vdash X \approx (a\,b){\cdot}X$$

Rules for Equivalence

$$\frac{\begin{array}{c}(a\ \#\ X) \in \nabla \\ \text{for all } a \text{ with } \pi{\cdot}a \neq \pi'{\cdot}a\end{array}}{\nabla \vdash \pi{\cdot}X \approx \pi'{\cdot}X}$$

for example

$$\{a \# X, c \# X\} \vdash (a\,c)(a\,b){\cdot}X \approx (b\,c){\cdot}X$$

because $(a\,c)(a\,b)$: $\begin{array}{l} a \mapsto b \\ b \mapsto c \\ c \mapsto a \end{array}$ $(b\,c)$: $\begin{array}{l} a \mapsto a \\ b \mapsto c \\ c \mapsto b \end{array}$

disagree at a and c.

Rules for Freshness

Excerpt
(again only the interesting rules)

Rules for Freshness

$$\frac{a \neq b}{\nabla \vdash a \# b}$$

$$\frac{}{\nabla \vdash a \# a.t}$$

$$\frac{a \neq b \qquad \nabla \vdash a \# t}{\nabla \vdash a \# b.t}$$

$$\frac{(\pi^{-1} \cdot a \# X) \in \nabla}{\nabla \vdash a \# \pi \cdot X}$$

$\approx$ is an Equivalence

Theorem: $\approx$ is an equivalence relation.

(Reflexivity) $\nabla \vdash t \approx t$

(Symmetry) if $\nabla \vdash t_1 \approx t_2$ then $\nabla \vdash t_2 \approx t_1$

(Transitivity) if $\nabla \vdash t_1 \approx t_2$ and $\nabla \vdash t_2 \approx t_3$
then $\nabla \vdash t_1 \approx t_3$

$\approx$ is an Equivalence

Theorem: $\approx$ is an equivalence relation.

because $\approx$ has very good properties:

- $\nabla \vdash t \approx t'$ then $\nabla \vdash \pi{\cdot}t \approx \pi{\cdot}t'$
- $\nabla \vdash a \# t$ then $\nabla \vdash \pi{\cdot}a \# \pi{\cdot}t$

$\approx$ is an Equivalence

Theorem: $\approx$ is an equivalence relation.

because $\approx$ has very good properties:

- $\nabla \vdash t \approx t'$ then $\nabla \vdash \pi{\cdot}t \approx \pi{\cdot}t'$
- $\nabla \vdash a \# t$ then $\nabla \vdash \pi{\cdot}a \# \pi{\cdot}t$
- $\nabla \vdash t \approx \pi{\cdot}t'$ then $\nabla \vdash (\pi^{-1}){\cdot}t \approx t'$
- $\nabla \vdash a \# \pi{\cdot}t$ then $\nabla \vdash (\pi^{-1}){\cdot}a \# t$

$\approx$ is an Equivalence

Theorem: $\approx$ is an equivalence relation.

because $\approx$ has very good properties:

- $\nabla \vdash t \approx t'$ then $\nabla \vdash \pi{\cdot}t \approx \pi{\cdot}t'$
- $\nabla \vdash a \# t$ then $\nabla \vdash \pi{\cdot}a \# \pi{\cdot}t$
- $\nabla \vdash t \approx \pi{\cdot}t'$ then $\nabla \vdash (\pi^{-1}){\cdot}t \approx t'$
- $\nabla \vdash a \# \pi{\cdot}t$ then $\nabla \vdash (\pi^{-1}){\cdot}a \# t$
- $\nabla \vdash a \# t$ and $\nabla \vdash t \approx t'$ then $\nabla \vdash a \# t'$

Comparison with $=_\alpha$

Traditionally $=_\alpha$ is defined as

least congruence which identifies $a.t$ with $b.[a := b]t$ provided b is not free in t

where $[a := b]t$ replaces all free occurrences of a by b in t.

Comparison with $=_\alpha$

Traditionally $=_\alpha$ is defined as

> least congruence which identifies $a.t$ with $b.[a := b]t$ provided b is not free in t

where $[a := b]t$ replaces all free occurrences of a by b in t.

For ground terms:

> <u>Theorem:</u> $t =_\alpha t'$ iff $\varnothing \vdash t \approx t'$
>
> $a \notin FA(t)$ iff $\varnothing \vdash a \# t$

Comparison with $=_\alpha$

Traditionally $=_\alpha$ is defined as

> least congruence which identifies $a.t$ with $b.[a := b]t$ provided b is not free in t

where $[a := b]t$ replaces all free occurrences of a by b in t.

In general $=_\alpha$ and $\approx$ are distinct!

> $a.X =_\alpha b.X$ but not
>
> $\varnothing \vdash a.X \approx b.X$ $(a \neq b)$

Comparison with $=_\alpha$

That is a crucial point: if we had

$$\varnothing \vdash a.X \approx b.X\,,$$

then applying $[X := a]$, $[X := b]$, ...
give two terms that are **not** α-equivalent.

The freshness constraints $a \# X$ and $b \# X$ rule out the problematic substitutions. Therefore

$$\{a \# X, b \# X\} \vdash a.X \approx b.X$$

does hold.

Substitutions

- $$\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$$
- $$\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$$

Substitutions

- $$\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$$
- $$\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$$

for example

$$a.(a\,b){\cdot}X \;\; [X := \langle b, Y\rangle]$$

Substitutions

- $\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$
- $\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$

for example

$$\underline{a.(a\,b){\cdot}X\ \ [X := \langle b, Y\rangle]}$$

$$\Rightarrow\quad a.\underline{(a\,b){\cdot}X[X := \langle b, Y\rangle]}$$

Substitutions

- $\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$
- $\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$

for example

$$
\begin{aligned}
& a.(a\,b){\cdot}X\ \ [X := \langle b, Y\rangle] \\
\Rightarrow\ \ & a.\underline{(a\,b){\cdot}X[X := \langle b, Y\rangle]} \\
\Rightarrow\ \ & a.(a\,b){\cdot}\underline{\langle b, Y\rangle}
\end{aligned}
$$

Substitutions

- $$\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$$

- $$\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$$

for example

$$
\begin{array}{rl}
 & a.(a\,b){\cdot}X \;\; [X := \langle b, Y\rangle] \\
\Rightarrow & a.(a\,b){\cdot}X[X := \langle b, Y\rangle] \\
\Rightarrow & a.\underline{(a\,b)}{\cdot}\langle b, Y\rangle \\
\Rightarrow & a.\langle a, (a\,b){\cdot}Y\rangle
\end{array}
$$

Substitutions

- $\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$
- $\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$
- if $\nabla \vdash t \approx t'$ and $\nabla' \vdash \sigma(\nabla)$
  then $\nabla' \vdash \sigma(t) \approx \sigma(t')$

Substitutions

- $\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$
- $\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$
- if $\nabla \vdash t \approx t'$ and $\nabla' \vdash \sigma(\nabla)$

  then $\nabla' \vdash \sigma(t) \approx \sigma(t')$

this means
$\nabla' \vdash a \# \sigma(X)$
holds for all
$(a \# X) \in \nabla$

Substitutions

- $\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$
- $\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$
- if $\nabla \vdash t \approx t'$ and $\nabla' \vdash \sigma(\nabla)$

  then $\nabla' \vdash \sigma(t) \approx \sigma(t')$

Substitutions

- $\sigma(a.t) \stackrel{\text{def}}{=} a.\sigma(t)$
- $\sigma(\pi{\cdot}X) \stackrel{\text{def}}{=} \begin{cases} \pi \cdot \sigma(X) & \text{if } \sigma(X) \neq X \\ \pi{\cdot}X & \text{otherwise} \end{cases}$
- if $\nabla \vdash t \approx t'$ and $\nabla' \vdash \sigma(\nabla)$
  then $\nabla' \vdash \sigma(t) \approx \sigma(t')$
- $\sigma(\pi{\cdot}t) = \pi{\cdot}\sigma(t)$

Equational Problems

An equational problem

$$t \approx ? \ t'$$

is solved by

- a substitution σ (terms for variables)
- and a set of freshness assumptions ∇

so that $\nabla \vdash \sigma(t) \approx \sigma(t')$.

Unifying equations may entail solving freshness problems.

E.g. assuming that $a \neq a'$, then

$$a.t \approx ?\; a'.t'$$

can only be solved if

$$t \approx ?\; (a\, a')\cdot t' \quad \text{and} \quad a \,\#?\; t'$$

can be solved.

Freshness Problems

A freshness problem

$$a \;\#?\; t$$

is solved by

- a substitution σ
- and a set of freshness assumptions ∇

so that $\nabla \vdash a \;\#\; \sigma(t)$.

Existence of MGUs

<u>Theorem</u>: there is an algorithm which, given a nominal unification problem P, decides whether or not it has a solution (σ, ∇), and returns a most *general* one if it does.

Existence of MGUs

Theorem: there is an algorithm which, given a nominal unification problem P, decides whether or not it has a solution (σ, ∇), and returns a most general one if it does.

straightforward definition:
"iff there exists a τ such that ..."

Existence of MGUs

Theorem: there is an algorithm which, given a nominal unification problem P, decides whether or not it has a solution (σ, ∇), and returns a most *general* one if it does.

Proof: one can reduce all the equations to 'solved form' first (creating a substitution), and then solve the freshness problems (easy).

Remember the Quiz?

Assuming that a and b are distinct variables, is it possible to find λ-terms M_1 to M_7 that make the following pairs α-equivalent?

- $\lambda a.\lambda b.(M_1\ b)$ and $\lambda b.\lambda a.(a\ M_1)$
- $\lambda a.\lambda b.(M_2\ b)$ and $\lambda b.\lambda a.(a\ M_3)$
- $\lambda a.\lambda b.(b\ M_4)$ and $\lambda b.\lambda a.(a\ M_5)$
- $\lambda a.\lambda b.(b\ M_6)$ and $\lambda a.\lambda a.(a\ M_7)$

If there is one solution for a pair, can you describe all its solutions?

Answers to the Quiz

$\lambda a.\lambda b.(M_1\, b)$ and $\lambda b.\lambda a.(a\, M_1)$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx? \, (a\,b)\cdot a.\langle a, M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx? \; b.\langle b, (a\,b){\cdot}M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \ \approx ?\ \ b.a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx ?\ b.\langle b, (a\,b)\cdot M_1\rangle\ ,\ \ a\ \#?\ a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle M_1, b\rangle \approx ?\ \langle b, (a\,b)\cdot M_1\rangle\ ,\ \ a\ \#?\ a.\langle a, M_1\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx? \; b.\langle b, (a\,b)\cdot M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle M_1, b\rangle \approx? \; \langle b, (a\,b)\cdot M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} M_1 \approx? \; b \;,\; b \approx? \; (a\,b)\cdot M_1 \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx? b.\langle b, (a\,b)\cdot M_1\rangle \;,\; a \#? a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle M_1, b\rangle \approx? \langle b, (a\,b)\cdot M_1\rangle \;,\; a \#? a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} M_1 \approx? b \;,\; b \approx? (a\,b)\cdot M_1 \;,\; a \#? a.\langle a, M_1\rangle$$

$$\stackrel{[M_1:=b]}{\Longrightarrow} b \approx? (a\,b)\cdot b \;,\; a \#? a.\langle a, b\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \ \approx? \ b.a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx? \ b.\langle b, (a\,b)\cdot M_1\rangle \ , \ a \ \#? \ a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle M_1, b\rangle \approx? \ \langle b, (a\,b)\cdot M_1\rangle \ , \ a \ \#? \ a.\langle a, M_1\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} M_1 \approx? \ b \ , \ b \approx? \ (a\,b)\cdot M_1 \ , \ a \ \#? \ a.\langle a, M_1\rangle$$

$$\stackrel{[M_1:=b]}{\Longrightarrow} b \approx? \ a \ , \ a \ \#? \ a.\langle a, b\rangle$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

$$\xRightarrow{\varepsilon} b.\langle M_1, b\rangle \approx? \; b.\langle b, (a\,b)\cdot M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\xRightarrow{\varepsilon} \langle M_1, b\rangle \approx? \; \langle b, (a\,b)\cdot M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\xRightarrow{\varepsilon} M_1 \approx? \; b \;,\; b \approx? \; (a\,b)\cdot M_1 \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\xRightarrow{[M_1:=b]} b \approx? \; a \;,\; a \;\#?\; a.\langle a, b\rangle$$

$$\Longrightarrow FAIL$$

Answers to the Quiz

$$a.b.\langle M_1, b\rangle \;\approx?\; b.a.\langle a, M_1\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b.\langle M_1, b\rangle \approx? b.\langle b, (a\,b)\cdot M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} \langle M_1, b\rangle \approx? \langle b, (a\,b)\cdot M_1\rangle \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} M_1 \approx? b \;,\; b \approx? (a\,b)\cdot M_1 \;,\; a \;\#?\; a.\langle a, M_1\rangle$$

$$\overset{[M_1:=b]}{\Longrightarrow} b \approx? a \;,\; a \;\#?\; a.\langle a, b\rangle$$

$$\Longrightarrow FAIL$$

$\lambda a.\lambda b.(M_1\, b) =_\alpha \lambda b.\lambda a.(a\, M_1)$ has no solution

Answers to the Quiz

$\lambda a.\lambda b.(b\, M_6)$ and $\lambda a.\lambda a.(a\, M_7)$

Answers to the Quiz

$$\boldsymbol{a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle}$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\; \approx ? \;\; a.a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx ? \; a.\langle a, M_7\rangle$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \,\#?\, \langle a, M_7\rangle$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \mathrel{\#?} \langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b \approx? b \;,\; M_6 \approx? (b\,a)\cdot M_7 \;,\; b \mathrel{\#?} \langle a, M_7\rangle$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \,\#?\, \langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b \approx? b \;,\; M_6 \approx? (b\,a)\cdot M_7 \;,\; b \,\#?\, \langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} M_6 \approx? (b\,a)\cdot M_7 \;,\; b \,\#?\, \langle a, M_7\rangle$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \#? \langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b \approx? b \;,\; M_6 \approx? (b\,a)\cdot M_7 \;,\; b \#? \langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} M_6 \approx? (b\,a)\cdot M_7 \;,\; b \#? \langle a, M_7\rangle$$

$$\overset{[M_6:=(b\,a)\cdot M_7]}{\Longrightarrow} b \#? \langle a, M_7\rangle$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \#? \langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} b \approx? b \;,\; M_6 \approx? (b\,a)\cdot M_7 \;,\; b \#? \langle a, M_7\rangle$$

$$\overset{\varepsilon}{\Longrightarrow} M_6 \approx? (b\,a)\cdot M_7 \;,\; b \#? \langle a, M_7\rangle$$

$$\overset{[M_6:=(b\,a)\cdot M_7]}{\Longrightarrow} b \#? \langle a, M_7\rangle$$

$$\overset{\varnothing}{\Longrightarrow} b \#? a \;,\; b \#? M_7$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \,\#?\, \langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b \approx? b \;,\; M_6 \approx? (b\,a)\cdot M_7 \;,\; b \,\#?\, \langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} M_6 \approx? (b\,a)\cdot M_7 \;,\; b \,\#?\, \langle a, M_7\rangle$$

$$\stackrel{[M_6:=(b\,a)\cdot M_7]}{\Longrightarrow} b \,\#?\, \langle a, M_7\rangle$$

$$\stackrel{\varnothing}{\Longrightarrow} b \,\#?\, a \;,\; b \,\#?\, M_7$$

$$\stackrel{\varnothing}{\Longrightarrow} b \,\#?\, M_7$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} \langle b, M_6\rangle \approx? \langle b, (b\,a)\cdot M_7\rangle \;,\; b \;\#?\; \langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} b \approx? b \;,\; M_6 \approx? (b\,a)\cdot M_7 \;,\; b \;\#?\; \langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow} M_6 \approx? (b\,a)\cdot M_7 \;,\; b \;\#?\; \langle a, M_7\rangle$$

$$\stackrel{[M_6:=(b\,a)\cdot M_7]}{\Longrightarrow} b \;\#?\; \langle a, M_7\rangle$$

$$\stackrel{\varnothing}{\Longrightarrow} b \;\#?\; a \;,\; b \;\#?\; M_7$$

$$\stackrel{\varnothing}{\Longrightarrow} b \;\#?\; M_7$$

$$\stackrel{\{b\#M_7\}}{\Longrightarrow} \varnothing$$

Answers to the Quiz

$$a.b.\langle b, M_6\rangle \;\approx?\; a.a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow}\; b.\langle b, M_6\rangle \approx? a.\langle a, M_7\rangle$$

$$\stackrel{\varepsilon}{\Longrightarrow}\; \langle b, M_6\rangle \approx$$

$$\stackrel{\varepsilon}{\Longrightarrow}\; b \approx? b \;,\; M$$

$$\stackrel{\varepsilon}{\Longrightarrow}\; M_6 \approx? (b$$

$$\stackrel{[M_6:=(b\,a)\cdot M_7]}{\Longrightarrow}\; b \;\#$$

$$\stackrel{\varnothing}{\Longrightarrow}\; b \;\#?\; a \;,\; b \;\#?\; M_7$$

$$\stackrel{\varnothing}{\Longrightarrow}\; b \;\#?\; M_7$$

$$\stackrel{\{b\#M_7\}}{\Longrightarrow}\; \varnothing$$

$$\lambda a.\lambda b.(b\, M_6) \;=_\alpha\; \lambda a.\lambda a.(a\, M_7)$$

we can take M_7 to be any λ-term that does not contain free occurrences of b, so long as we take M_6 to be the result of swapping all occurrences of b and a throughout M_7

Conclusion

- used a permutation operation for renaming (has much nicer properties)

Conclusion

- used a permutation operation for renaming (has much nicer properties) !!!

Conclusion

- used a permutation operation for renaming (has much nicer properties) !!!
- have concrete names for binders (nominal unification) and **not** de-Bruijn indices

Conclusion

- used a permutation operation for renaming (has much nicer properties) !!!
- have concrete names for binders (nominal unification) and **not** de-Bruijn indices
- it is a completely first-order language

Conclusion

- used a permutation operation for renaming (has much nicer properties) !!!
- have concrete names for binders (nominal unification) and **not** de-Bruijn indices
- it is a completely first-order language
- computed with freshness assumptions; this allowed us to define $\approx$ so that substitution respects α-equivalence

Conclusion

- used a permutation operation for renaming (has much nicer properties) !!!
- have concrete names for binders (nominal unification) and **not** de-Bruijn indices
- it is a completely first-order language
- computed with freshness assumptions; this allowed us to define $\approx$ so that substitution respects α-equivalence
- verified everything in Isabelle

Is it useful?

- applications to logic programming (w. J. Cheney)

$$\frac{x : A \in \Gamma}{\Gamma \triangleright x : A} \qquad \frac{\Gamma \triangleright M : A \supset B \quad \Gamma \triangleright N : A}{\Gamma \triangleright M\,N : B} \qquad \frac{x : A, \Gamma \triangleright M : B}{\Gamma \triangleright \lambda x.M : A \supset B}$$

Is it useful?

- applications to logic programming (w. J. Cheney)

$$\frac{x : A \in \Gamma}{\Gamma \triangleright x : A} \qquad \frac{\Gamma \triangleright M : A \supset B \quad \Gamma \triangleright N : A}{\Gamma \triangleright M\,N : B} \qquad \frac{x : A, \Gamma \triangleright M : B}{\Gamma \triangleright \lambda x.M : A \supset B}$$

```
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) / x#Gamma :-
                          type (pair x A)::Gamma M B.

member A A::Tail.

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

Is it useful?

- applications to logic programming (w. J. Cheney)

$$\frac{x : A \in \Gamma}{\Gamma \rhd x : A} \qquad \frac{\Gamma \rhd M : A \supset B \quad \Gamma \rhd N : A}{\Gamma \rhd M\, N : B} \qquad \frac{x : A, \Gamma \rhd M : B}{\Gamma \rhd \lambda x.M : A \supset B}$$

- term-rewriting (Knuth-Bendix)

  Roughly: given a rewrite system, which reduction need to be added in order to get confluence.

  No such algorithm for rewriting with binders.

The End

Paper and Isabelle scripts at:

www.cl.cam.ac.uk/~cu200/Unification

Most General Unifiers

Definition: for a unification problem P, a solution (σ_1, ∇_1) is more general than another solution (σ_2, ∇_2), iff there exists a substitution σ with

- $\nabla_2 \vdash \sigma(\nabla_1)$
- $\nabla_2 \vdash \sigma_2 \approx \sigma \circ \sigma_1$

Most General Unifiers

Definition: for a unification problem P, a solution (σ_1, ∇_1) is more general than another solution (σ_2, ∇_2), iff there exists a substitution σ with

- $\nabla_2 \vdash \sigma(\nabla_1)$

  $\nabla_2 \vdash a \# \sigma(X)$ holds for all $(a \# X) \in \nabla_1$

- $\nabla_2 \vdash \sigma_2 \approx \sigma \circ \sigma_1$

Most General Unifiers

Definition: for a unification problem P, a solution (σ_1, ∇_1) is more general than another solution (σ_2, ∇_2), iff there exists a substitution σ wit

$\nabla_2 \vdash \sigma_2(X) \approx \sigma(\sigma_1(X))$ holds for all $X \in \mathrm{dom}(\sigma_2) \cup \mathrm{dom}(\sigma \circ \sigma_1)$

- $\nabla_2 \vdash \sigma(\nabla_1)$
- $\nabla_2 \vdash \sigma_2 \approx \sigma \circ \sigma_1$