# Quiz

Assuming that $a$ and $b$ are distinct variables, is it possible to find $\lambda$-terms $M_1$ to $M_7$ that make the following pairs $\alpha$-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 $L_\lambda$ 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:

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

# ...and Substitution

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

for example:

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

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

# ...and Substitution

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

for example:

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

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:

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

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

# ...and Substitution

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

for example:

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

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

Does it have to be so? No!

# Swappings

Problem: substitution does not respect $\alpha$-equivalence, e.g.

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

# Swappings

Problem: substitution does not respect $\alpha$-equivalence, e.g.

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

# Swappings

Problem: substitution does not respect $\alpha$-equivalence, e.g.

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

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

# Swappings

Problem: substitution does not respect $\alpha$-equivalence, e.g.

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

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 $\alpha$-equivalence, e.g.

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

Nice Alternative: use a less complicated operation for renaming

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

be they free, bound or binding

# Swappings

Problem: substitution does not respect $\alpha$-equivalence, e.g.

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

<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 $\alpha$-equivalence, e.g.

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

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

# Terms

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

# Terms

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

$\ulcorner \lambda a.a \urcorner \mapsto \texttt{fn}\; a.a$

constructions like $\texttt{fn}\; X.X$ are not allowed

# Terms

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

# Terms

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

$\pi$ 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

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

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

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

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

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

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

# Freshness Assumptions

Our equality is **not** just

$$t \approx t'$$ $\alpha$-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

$$\nabla \vdash t \approx t'$$ α-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.

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

# 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

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

# Rules for Equivalence

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

for example

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

# Rules for Equivalence

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

for example

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

because $(a\,c)(a\,b)$: $a \mapsto b$, $b \mapsto c$, $c \mapsto a$ $\quad (b\,c)$: $a \mapsto a$, $b \mapsto c$, $c \mapsto b$

disagree at $a$ and $c$.

# Rules for Freshness

Excerpt
(again only the interesting rules)

# Rules for Freshness

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

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

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

# $\approx$ 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:

Theorem: $t =_\alpha t'$ iff $\varnothing \vdash t \approx t'$

$a \not\in FA(t)$ 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** $\alpha$-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

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

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

$$\Rightarrow \quad a.(a\,b){\cdot}\underline{\langle b, Y\rangle}$$

# 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

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

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

$$\Rightarrow\ a.\underline{(a\,b)}{\cdot}\langle b, Y\rangle$$

$$\Rightarrow\ a.\langle a, (a\,b){\cdot}Y\rangle$$

# 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 $\sigma$ (terms for variables)
- and a set of freshness assumptions $\nabla$

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 $\sigma$
- and a set of freshness assumptions $\nabla$

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

# Existence of MGUs

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

straightforward definition:
"iff there exists a $\tau$ 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 $(\sigma, \nabla)$, 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 $\lambda$-terms $M_1$ to $M_7$ that make the following pairs $\alpha$-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

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

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

# Answers to the Quiz

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

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

# 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

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

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

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

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

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

# Answers to the Quiz

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

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

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

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

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

# Answers to the Quiz

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

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

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

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

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

$$\Longrightarrow FAIL$$

# Answers to the Quiz

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

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

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

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

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

$$\Longrightarrow FAIL$$

$\lambda a.\lambda b.(M_1\, b) =_\alpha \lambda b.\lambda a.(a\, M_1)$ 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

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

# Answers to the Quiz

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

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

# 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

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

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

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

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

# Answers to the Quiz

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

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

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

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

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

# Answers to the Quiz

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

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

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

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

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

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

# Answers to the Quiz

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

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

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

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

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

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

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

# 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

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

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

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

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

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

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

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

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

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

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

we can take $M_7$ to be any $\lambda$-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 $\alpha$-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 $\alpha$-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)

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

```
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)

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

- 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 $\boldsymbol{P}$, a solution $(\boldsymbol{\sigma_1}, \boldsymbol{\nabla_1})$ is more general than another solution $(\boldsymbol{\sigma_2}, \boldsymbol{\nabla_2})$, iff there exists a substitution $\boldsymbol{\sigma}$ with

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

# Most General Unifiers

<u>Definition</u>: for a unification problem $P$, a solution $(\sigma_1, \nabla_1)$ is more general than another solution $(\sigma_2, \nabla_2)$, iff there exists a substitution $\sigma$ 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 $(\sigma_1, \nabla_1)$ is more general than another solution $(\sigma_2, \nabla_2)$, iff there exists a substitution $\sigma$ wit

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

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