A Relational Logic for Higher-Order Programs

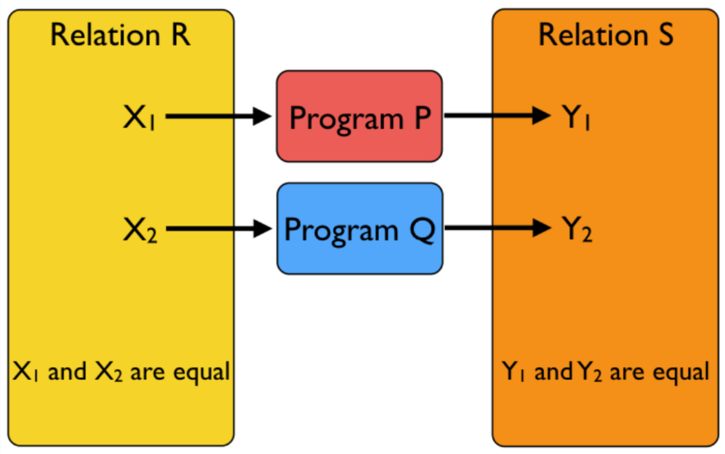
Alejandro Aguirre[1] Gilles Barthe[1] Marco Gaboardi[2] Deepak Garg[3] Pierre-Yves Strub[4]

ICFP, Sep 5 2017

[1]Imdea Software; [2]University at Buffalo, SUNY; [3]MPI-SWS; [4]École Polytechnique

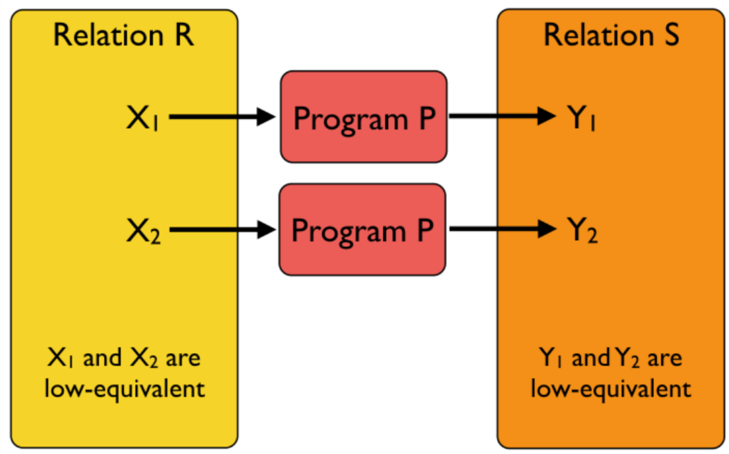
Relational properties a.k.a. 2-properties (I)

Two runs of two programs, e.g. equivalence...



Relational properties a.k.a. 2-properties (II)

...or two runs of the same program, e.g. non-interference



Relational refinement types (I)

Refinement types extend types with logical properties:

$$\Gamma \vdash t : \{x : \mathbb{N} \mid \exists z. x = 2 * z\}$$

Relational refinement types [1] generalize them to a relational setting:

$$\Gamma \vdash t_1 \sim t_2 : \{x : \mathbb{N} \mid x_1 = x_2\}$$

[1] Gilles Barthe, Cédric Fournet, Benjamin Grégoire, Pierre-Yves Strub, Nikhil Swamy, and Santiago Zanella Béguelin. *Probabilistic relational verification for cryptographic implementations* (POPL '14)

Relational refinement types (II)

Pros:

- Very intuitive (e.g. $\{x : \mathbb{N} \mid x_1 \leq x_2\} \to \{y : \mathbb{N} \mid y_1 \leq y_2\}$ types monotonic functions)
- Syntax directed
- Exploit structural similarities

$$\text{if } b \text{ then } 2{*}x \text{ else } x{+}1 \ \sim\ \text{if } b \text{ then } 2{*}x \text{ else } x{-}1$$

- Lots of theoretical and practical developments for unary refinements could be reused

Limits of RRT

We want to prove:

$$\text{take } n \; (\text{map } f \; l) = \text{map } f \; (\text{take } n \; l)$$

Limits of RRT

We want to prove:

$$\mathrm{take}\ n\ (\mathrm{map}\ f\ l) = \mathrm{map}\ f\ (\mathrm{take}\ n\ l)$$

Which type to give it?

$$\mathrm{take}\ n\ (\mathrm{map}\ f\ l) \sim \mathrm{map}\ f\ (\mathrm{take}\ n\ l) : \{x : \mathsf{list}_{\mathbb{N}} \mid x_1 = x_2\}$$

Limits of RRT

We want to prove:

$$\text{take } n \; (\text{map } f \; l) = \text{map } f \; (\text{take } n \; l)$$

Which type to give it?

$$\text{take } n \; (\text{map } f \; l) \sim \text{map } f \; (\text{take } n \; l) : \{x : \mathsf{list}_{\mathbb{N}} \mid x_1 = x_2\}$$

We apply [APP] rule...

$$\text{take} \sim \text{map} : \{x :? \mid ?\}$$

Our contributions

- A foundational system to prove relational properties
- in a syntax directed way
- not restricted by types or structure

Base logic: HOL

- λ-terms over simple + inductive types
- (Axiomatically defined) Predicates: $P(t_1, \ldots, t_n)$

$$\forall l. prefix([], l) \quad \forall xtl. prefix(t, l) \Rightarrow prefix(x :: t, x :: l)$$

- Propositional connectives: $\wedge, \vee, \Rightarrow$
- Quantification over simple/inductive types: $\forall(x : \tau)$, $\exists(x : \tau)$

Base logic: HOL

- λ-terms over simple + inductive types
- (Axiomatically defined) Predicates: $P(t_1, \ldots, t_n)$

$$\forall l.\mathit{prefix}([], l) \quad \forall x t l.\mathit{prefix}(t, l) \Rightarrow \mathit{prefix}(x :: t, x :: l)$$

- Propositional connectives: $\wedge, \vee, \Rightarrow$
- Quantification over simple/inductive types: $\forall(x : \tau)$, $\exists(x : \tau)$

Why not just use this?

Base logic: HOL

- λ-terms over simple + inductive types
- (Axiomatically defined) Predicates: $P(t_1, \ldots, t_n)$

$$\forall l.prefix([], l) \quad \forall xtl.prefix(t, l) \Rightarrow prefix(x :: t, x :: l)$$

- Propositional connectives: $\wedge, \vee, \Rightarrow$
- Quantification over simple/inductive types: $\forall (x : \tau)$, $\exists (x : \tau)$

Why not just use this?

No syntax directedness or structural reasoning

Relational HOL

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

Context

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Relational HOL

Judgements combine **types** and **logic**:

Assertions over Γ

$$\text{HOL:} \quad \boxed{\Gamma} \mid \boxed{\Psi} \vdash \phi$$

Context

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Judgements combine **types** and **logic**:

Assertions over Γ

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

Context

Predicate

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Relational HOL

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash \boxed{t_1 : \tau_1} \mid \phi(\mathbf{r})$$

Term

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Relational HOL

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash \underbrace{t_1 : \tau_1}_{\text{Term}} \mid \underbrace{\phi(\mathbf{r})}_{\text{Predicate}}$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Relational HOL

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash \boxed{t_1 : \tau_1} \sim t_2 : \tau_2 \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

1st term

Relational HOL

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash \underbrace{t_1 : \tau_1}_{\text{1st term}} \sim \underbrace{t_2 : \tau_2}_{\text{2nd term}} \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Relational HOL

Judgements combine **types** and **logic**:

$$\text{HOL:} \quad \Gamma \mid \Psi \vdash \phi$$

$$\text{UHOL:} \quad \Gamma \mid \Psi \vdash t_1 : \tau_1 \mid \phi(\mathbf{r})$$

$$\text{RHOL:} \quad \Gamma \mid \Psi \vdash \underbrace{t_1 : \tau_1}_{\text{1st term}} \sim \underbrace{t_2 : \tau_2}_{\text{2nd term}} \mid \underbrace{\phi(\mathbf{r}_1, \mathbf{r}_2)}_{\text{Predicate}}$$

Refinement types vs RHOL

Key Idea: separation of concerns between types and assertions

- Unary

$$\vdash t : \{x : \tau \mid \phi(x)\} \longrightarrow t : \tau \mid \phi(\mathbf{r})$$

- Relational

$$t_1 \sim t_2 : \{x : \tau \mid \phi(x_1, x_2)\} \longrightarrow t_1 : \tau \sim t_2 : \tau \mid \phi(\mathbf{r}_1, \mathbf{r}_2)$$

Two-sided and one-sided rules

Two-sided rules relate two terms with the same top term former

$$\lambda x_1.t_1 \sim \lambda x_2.t_2$$

One-sided rules relate two terms with *different* top term former

$$\lambda x_1.t_1 \sim t_2\ \ u_2$$

Two-sided rules

Judgements combine **types** and **logic**:

Abstraction

$$\frac{\Gamma, x_1 : \tau_1, x_2 : \tau_2 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1.t_1 : \tau_1 \to \sigma_1 \sim \lambda x_2.t_2 : \tau_2 \to \sigma_2 \mid \forall x_1, x_2.\phi' \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1][\mathbf{r}_2\ x_2/\mathbf{r}_2]}$$

Two-sided rules

Judgements combine **types** and **logic**:

Abstraction

$$\frac{\Gamma, x_1 : \tau_1, x_2 : \tau_2 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1.t_1 : \tau_1 \to \sigma_1 \sim \lambda x_2.t_2 : \tau_2 \to \sigma_2 \mid \forall x_1, x_2.\phi' \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1][\mathbf{r}_2\ x_2/\mathbf{r}_2]}$$

Two-sided rules

Judgements combine **types** and **logic**:

Abstraction

$$\frac{\Gamma, x_1 : \tau_1, x_2 : \tau_2 \mid \boldsymbol{\Psi}, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \boldsymbol{\Psi} \vdash \lambda x_1.t_1 : \tau_1 \to \sigma_1 \sim \lambda x_2.t_2 : \tau_2 \to \sigma_2 \mid \forall x_1, x_2.\phi' \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1][\mathbf{r}_2\ x_2/\mathbf{r}_2]}$$

Two-sided rules

Judgements combine **types** and **logic**:

Abstraction

$$\frac{\Gamma, x_1 : \tau_1, x_2 : \tau_2 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1.t_1 : \tau_1 \to \sigma_1 \sim \lambda x_2.t_2 : \tau_2 \to \sigma_2 \mid \forall x_1, x_2.\phi' \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1][\mathbf{r}_2\ x_2/\mathbf{r}_2]}$$

Two-sided rules

Judgements combine **types** and **logic**:

Abstraction

$$\frac{\Gamma, x_1 : \tau_1, x_2 : \tau_2 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1.t_1 : \tau_1 \to \sigma_1 \sim \lambda x_2.t_2 : \tau_2 \to \sigma_2 \mid \forall x_1, x_2.\phi' \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1][\mathbf{r}_2\ x_2/\mathbf{r}_2]}$$

Application

$$\frac{\begin{array}{c}\Gamma \mid \Psi \vdash t_1 : \tau_1 \to \sigma_1 \sim t_2 : \tau_2 \to \sigma_2 \mid \forall x_1, x_2.\phi'[x_1/\mathbf{r}_1][x_2/\mathbf{r}_2] \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1][\mathbf{r}_2\ x_2/\mathbf{r}_2] \\ \Gamma \mid \Psi \vdash u_1 : \tau_1 \sim u_2 : \tau_2 \mid \phi'\end{array}}{\Gamma \mid \Psi \vdash t_1 u_1 : \sigma_1 \sim t_2 u_2 : \sigma_2 \mid \phi[u_1/x_1][u_2/x_2]}$$

One-sided rules

Abstraction

$$\frac{\Gamma, x_1 : \tau_1 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1 . t_1 : \tau_1 \to \sigma_1 \sim t_2 : \sigma_2 \mid \forall x_1 . \phi' \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1]}$$

Application

$$\frac{\Gamma \mid \Psi \vdash t_1 : \tau_1 \to \sigma_1 \sim u_2 : \sigma_2 \mid \forall x_1 . \phi'[x_1/\mathbf{r}_1] \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1] \qquad \Gamma \mid \Psi \vdash u_1 : \sigma_1 \mid \phi'}{\Gamma \mid \Psi \vdash t_1 u_1 : \sigma_1 \sim u_2 : \sigma_2 \mid \phi[u_1/x_1]}$$

One-sided rules

Abstraction

$$\frac{\Gamma, x_1 : \tau_1 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1 . t_1 : \tau_1 \to \sigma_1 \sim t_2 : \sigma_2 \mid \forall x_1 . \phi' \Rightarrow \phi[\mathbf{r}_1 \ x_1 / \mathbf{r}_1]}$$

Application

$$\frac{\begin{array}{c}\Gamma \mid \Psi \vdash t_1 : \tau_1 \to \sigma_1 \sim u_2 : \sigma_2 \mid \forall x_1 . \phi'[x_1/\mathbf{r}_1] \Rightarrow \phi[\mathbf{r}_1 \ x_1/\mathbf{r}_1] \\ \Gamma \mid \Psi \vdash u_1 : \sigma_1 \mid \phi'\end{array}}{\Gamma \mid \Psi \vdash t_1 u_1 : \sigma_1 \sim u_2 : \sigma_2 \mid \phi[u_1/x_1]}$$

One-sided rules

Abstraction

$$\frac{\Gamma, x_1 : \tau_1 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1 . t_1 : \tau_1 \to \sigma_1 \sim t_2 : \sigma_2 \mid \forall x_1 . \phi' \Rightarrow \phi[\mathbf{r}_1\ x_1 / \mathbf{r}_1]}$$

Application

$$\frac{\begin{array}{c}\Gamma \mid \Psi \vdash t_1 : \tau_1 \to \sigma_1 \sim u_2 : \sigma_2 \mid \forall x_1 . \phi'[x_1/\mathbf{r}_1] \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1] \\ \Gamma \mid \Psi \vdash u_1 : \sigma_1 \mid \phi'\end{array}}{\Gamma \mid \Psi \vdash t_1 u_1 : \sigma_1 \sim u_2 : \sigma_2 \mid \phi[u_1/x_1]}$$

One-sided rules

Abstraction

$$\frac{\Gamma, x_1 : \tau_1 \mid \Psi, \phi' \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}{\Gamma \mid \Psi \vdash \lambda x_1 . t_1 : \tau_1 \to \sigma_1 \sim t_2 : \sigma_2 \mid \forall x_1 . \phi' \Rightarrow \phi[\mathbf{r}_1\ x_1 / \mathbf{r}_1]}$$

Application

$$\frac{\Gamma \mid \Psi \vdash t_1 : \tau_1 \to \sigma_1 \sim u_2 : \sigma_2 \mid \forall x_1 . \phi'[x_1/\mathbf{r}_1] \Rightarrow \phi[\mathbf{r}_1\ x_1/\mathbf{r}_1] \qquad \Gamma \mid \Psi \vdash u_1 : \sigma_1 \mid \phi'}{\Gamma \mid \Psi \vdash t_1 u_1 : \sigma_1 \sim u_2 : \sigma_2 \mid \phi[u_1/x_1]}$$

The SUB rule

Allows us to fall back to HOL:

$$\frac{\Gamma \mid \Psi \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi' \quad \Gamma \mid \Psi \vdash_{\mathsf{HOL}} \phi'[t_1/\mathbf{r}_1][t_2/\mathbf{r}_2] \Rightarrow \phi[t_1/\mathbf{r}_1][t_2/\mathbf{r}_2]}{\Gamma \mid \Psi \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi}$$

Equivalence between HOL and RHOL

$$\Gamma \mid \Psi \vdash t_1 : \sigma_1 \sim t_2 : \sigma_2 \mid \phi$$
$$\Updownarrow$$
$$\Gamma \mid \Psi \vdash \phi[t_1/\mathbf{r}_1][t_2/\mathbf{r}_2]$$

Plus: subject reduction, soundness, ...

Embeddings

RHOL is also useful as a framework in which to embed other relational typing systems:

- Relational Refinement Types
- DCC (dependency)
- RelCost (relational cost)

We get "for free" proofs of soundness.

Since RHOL is more expressive, we *can verify new examples*.

$$|s| = |t|, sorted(s) \vdash isort\ s \sim isort\ t \mid cost\ \mathbf{r}_1 \leq cost\ \mathbf{r}_2$$

Conclusions

- Relational refinement types have limited one-sided reasoning
- HOL is expressive but does not exploit structural similarities
- RHOL combines the best of both worlds in a lossless way: expressiveness + two-sided reasoning + 1 sided-reasoning
- This makes RHOL a foundational system
- Future work: extension with effects & implementation

Conclusions

- Relational refinement types have limited one-sided reasoning
- HOL is expressive but does not exploit structural similarities
- RHOL combines the best of both worlds in a lossless way: expressiveness + two-sided reasoning + 1 sided-reasoning
- This makes RHOL a foundational system
- Future work: extension with effects & implementation

Thanks!

Questions?

Embedding refinement types

We can embed refinement types into our system:

$$\llbracket \{y : \tau \mid \phi\} \rrbracket(x_1, x_2) \triangleq \bigwedge_{i \in \{1,2\}} \lfloor \tau \rfloor (x_i) \wedge \phi[x_i/y]$$

$$\llbracket \{y :: T \mid \phi\} \rrbracket(x_1, x_2) \triangleq \llbracket T \rrbracket(x_1, x_2) \wedge \phi[x_1/y_1][x_2/y_2]$$

$$\llbracket \Pi(y : \tau).\sigma \rrbracket(x) \triangleq \bigwedge_{i \in \{1,2\}} \forall y. \lfloor \tau \rfloor (y) \Rightarrow \lfloor \sigma \rfloor (xy)$$

$$\llbracket \Pi(y :: T).\, U \rrbracket(x_1, x_2) \triangleq \forall y_1 y_2. \llbracket T \rrbracket(y_1, y_2) \Rightarrow \llbracket \sigma \rrbracket(x_1 y_1,\ x_2 y_2)$$

Example: factorial (I)

We can implement factorial without and with accumulator:

$$\mathrm{fact}_1 \equiv \mathrm{letrec}\ f_1\ x_1\ =\ \mathrm{case}\ x_1\ \mathrm{of}\ [0 \to 1; Sy_1 \to (Sy_1) * (f_1 y_1)]$$

$$\mathrm{fact}_2 \equiv \mathrm{letrec}\ f_2\ x_2 = \lambda a.\mathrm{case}\ x_2\ \mathrm{of}\ [0 \to a; Sy_2 \to f_2\ y_2\ (a*(Sy_2))]$$

We want to prove:

$$\emptyset \mid \emptyset \vdash \mathrm{fact}_1 \sim \mathrm{fact}_2 \mid \forall x_1 x_2 a.\ x_1 = x_2 \Rightarrow (r_1\ x_1) * a = r_2\ x_2\ a$$

Notice that the two programs have different types: $\mathbb{N} \to \mathbb{N}$ and $\mathbb{N} \to \mathbb{N} \to \mathbb{N}$

Example: factorial (II)

Proof reduces to:

$$\Gamma \mid \psi \vdash \text{case } x_1 \text{ of } [0 \to 1; Sy_1 \to (Sy_1) * (f_1 y_1)] \sim$$
$$\text{case } x_2 \text{ of } [0 \to a; Sy_2 \to f_2\ y_2\ (a * (Sy_2))] \mid \mathbf{r}_1 * a = \mathbf{r}_2$$

where ψ is the “inductive hypothesis”

Example: factorial (II)

Proof reduces to:

$$\Gamma \mid \psi \vdash \mathrm{case}\ x_1\ \mathrm{of}\ [0 \to 1; Sy_1 \to (Sy_1) * (f_1 y_1)] \sim$$
$$\mathrm{case}\ x_2\ \mathrm{of}\ [0 \to a; Sy_2 \to f_2\ y_2\ (a * (Sy_2))] \mid \mathbf{r}_1 * a = \mathbf{r}_2$$

where ψ is the "inductive hypothesis"

Proof obligations:

- $\Gamma \mid \psi \vdash 1 \sim a \mid r_1 * a = r_2$

Example: factorial (II)

Proof reduces to:

$$\Gamma \mid \psi \vdash \mathrm{case}\ x_1\ \mathrm{of}\ [0 \to 1; Sy_1 \to (Sy_1) * (f_1 y_1)] \sim \\ \mathrm{case}\ x_2\ \mathrm{of}\ [0 \to a; Sy_2 \to f_2\ y_2\ (a * (Sy_2))] \mid \mathbf{r}_1 * a = \mathbf{r}_2$$

where ψ is the "inductive hypothesis"

Proof obligations:

- $\Gamma \mid \psi \vdash 1 \sim a \mid r_1 * a = r_2$

Trivial

Example: factorial (II)

Proof reduces to:

$$\Gamma \mid \psi \vdash \text{case } x_1 \text{ of } [0 \to 1; Sy_1 \to (Sy_1) * (f_1 y_1)] \sim \\ \text{case } x_2 \text{ of } [0 \to a; Sy_2 \to f_2\ y_2\ (a * (Sy_2))] \mid \mathbf{r}_1 * a = \mathbf{r}_2$$

where ψ is the "inductive hypothesis"

Proof obligations:

- $\Gamma \mid \psi \vdash 1 \sim a \mid r_1 * a = r_2$
- $\Gamma \mid \psi, x_1 = Sy_2, x_2 = Sy_2 \vdash (Sy_1) * (f_1 y_1) \sim f_2\ y_2\ (a * (Sy_2)) \mid (r_1\ y_1) * a = (r_2\ y_2)$

Example: factorial (II)

Proof reduces to:

$$\Gamma \mid \psi \vdash \text{case } x_1 \text{ of } [0 \to 1; Sy_1 \to (Sy_1) * (f_1 y_1)] \sim \\ \text{case } x_2 \text{ of } [0 \to a; Sy_2 \to f_2\ y_2\ (a * (Sy_2))] \mid \mathbf{r}_1 * a = \mathbf{r}_2$$

where ψ is the "inductive hypothesis"

Proof obligations:

- $\Gamma \mid \psi \vdash 1 \sim a \mid r_1 * a = r_2$
- $\Gamma \mid \psi, x_1 = Sy_2, x_2 = Sy_2 \vdash (Sy_1) * (f_1 y_1) \sim f_2\ y_2\ (a * (Sy_2)) \mid (r_1\ y_1) * a = (r_2\ y_2)$

By instantiating ψ