Subject Reduction for Synchronous MPST in Coq

Part II: Subject Reduction Proof

November 20, 2024

Outline

Theorem

Have

- $\vdash \mathcal{M} : \mathsf{G}$
- $\mathcal{M} \rightarrow^{\star}_{\beta} \mathcal{M}'$

Then

- $\exists\, \mathsf{G}'$ such that $\vdash \mathcal{M}' : \mathsf{G}'$ and $\mathsf{G} \Longrightarrow \mathsf{G}'$

Global Types

Global types are inductively generated by the following grammar.

$$
\begin{array}{lcl}
\mathsf{S} & ::= & \mathtt{nat} \;\mid\; \mathtt{int} \;\mid\; \mathtt{bool} \\
\mathbb{G} & ::= & \mathbf{end} \;\mid\; \mu\mathbf{t}.\mathbb{G} \;\mid\; \mathbf{t} \;\mid \\
 & & \mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathbb{G}_i\}_{i\in I}
\end{array}
$$

```
Inductive global: Type ≜
  | g_var : fin → global
  | g_end : global
  | g_send: part → part →
            list (option(sort*global)) → global
  | g_rec : global → global.
```

Global Types

Global types are inductively generated by the following grammar.

$$
\begin{array}{lclclclc}
\mathsf{S} & ::= & \mathtt{nat} & \mid & \mathtt{int} & \mid & \mathtt{bool} & \\
\mathbb{G} & ::= & \mathbf{end} & \mid & \mu\mathbf{t}.\mathbb{G} & \mid & \mathbf{t} & \mid \\
& & \multicolumn{6}{l}{\mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathbb{G}_i\}_{i\in I}}
\end{array}
$$

```
Inductive global: Type ≜
  | g_var : fin → global
  | g_end : global
  | g_send: part → part →
            list (option(sort*global)) → global
  | g_rec : global → global.
```

Global Type Trees

Global type trees are coinductively generated by the following grammar.

$$
\mathsf{G} \quad ::= \quad \mathbf{end} \quad \mid \quad \mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i\in I}
$$

```
CoInductive gtt: Type ≜
  | gtt_end : gtt
  | gtt_send: part → part →
              list (option(sort*gtt)) → gtt.
```

Global Types → Global Type Trees

- Recursive unfoldings are mapped to the same tree

$$\dfrac{\forall i \in I, \quad \mathbb{G}_i \xrightarrow{\mathcal{G}} \mathsf{G}_i}{\mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathbb{G}_i\}_{i \in I} \xrightarrow{\mathcal{G}} \mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i \in I}} \text{[gtrans-send]} \qquad \dfrac{}{\mathbf{end} \xrightarrow{\mathcal{G}} \mathbf{end}} \text{[gtrans-end]}$$

$$\dfrac{\mu \mathbf{t}.\mathbb{G} \xrightarrow{\mathcal{G}} \mathsf{G}}{\mathbb{G}[\mu \mathbf{t}.\mathbb{G}/\mathbf{t}] \xrightarrow{\mathcal{G}} \mathsf{G}} \text{[gtrans-rec]}$$

Global Types $\to$ Global Type Trees

- Recursive unfoldings are mapped to the same tree

$$\dfrac{\forall i \in I, \quad \mathbb{G}_i \xrightarrow{\mathcal{G}} \mathsf{G}_i}{\mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathbb{G}_i\}_{i\in I} \xrightarrow{\mathcal{G}} \mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i\in I}}\ [\text{gtrans-send}] \qquad \dfrac{}{\mathbf{end} \xrightarrow{\mathcal{G}} \mathbf{end}}\ [\text{gtrans-end}]$$

$$\dfrac{\mu\mathbf{t}.\mathbb{G} \xrightarrow{\mathcal{G}} \mathsf{G}}{\mathbb{G}[\mu\mathbf{t}.\mathbb{G}/\mathbf{t}] \xrightarrow{\mathcal{G}} \mathsf{G}}\ [\text{gtrans-rec}]$$

Example

$$\mathbb{G} = \mu\mathbf{t}.\mathsf{p} \to \mathsf{q} \begin{cases} \ell_1(\texttt{bool}).\mathbf{t} \\ \ell_2(\texttt{nat}).\mathbf{end} \end{cases}$$

Global Types $\to$ Global Type Trees

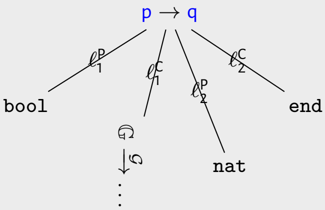
- Recursive unfoldings are mapped to the same tree

$$\frac{\forall i \in I, \quad \mathbb{G}_i \xrightarrow{\mathcal{G}} \mathsf{G}_i}{\mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathbb{G}_i\}_{i\in I} \xrightarrow{\mathcal{G}} \mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i\in I}}\ [\text{gtrans-send}] \qquad \frac{}{\mathtt{end} \xrightarrow{\mathcal{G}} \mathtt{end}}\ [\text{gtrans-end}]$$

$$\frac{\mu\mathbf{t}.\mathbb{G} \xrightarrow{\mathcal{G}} \mathsf{G}}{\mathbb{G}[\mu\mathbf{t}.\mathbb{G}/\mathbf{t}] \xrightarrow{\mathcal{G}} \mathsf{G}}\ [\text{gtrans-rec}]$$

Example

$$\mathbb{G} = \mu\mathbf{t}.\mathsf{p} \to \mathsf{q} \begin{cases} \ell_1(\mathtt{bool}).\mathbf{t} \\ \ell_2(\mathtt{nat}).\mathtt{end} \end{cases} \qquad \xrightarrow{\mathcal{G}}$$



Global Types → Global Type Trees

```
Inductive gttT (R : global → gtt → Prop) : global → gtt → Prop ≜
  | ...
  | gttT_rec: ∀ G Q G', R Q G' → subst_global 0 0 (g_rec G) G Q → gttT R (g_rec G) G'.

Definition gttTC G G' ≜ paco2 gttT bot2 G G'.
```

Local Types

Local types are inductively generated by the following grammar.

$$\mathbb{T} \quad ::= \quad \mathbf{end} \quad | \quad \mu\mathbf{t}.\mathbb{T} \quad | \quad \mathbf{t} \quad | \quad \bigoplus_{i\in I} \mathsf{p}!\ell_i(\mathsf{S}_i).\mathbb{T}_i \quad | \quad \&_{i\in I} \mathsf{p}?\ell_i(\mathsf{S}_i).$$

```
Inductive local  : Type ≜
  | l_var : fin → local
  | l_end : local
  | l_send: part →
            list (option(sort*local)) → local
  | l_recv: part →
            list (option(sort*local)) → local
  | l_rec : local → local.
```

Local Types

Local types are inductively generated by the following grammar.

$$\mathbb{T} \quad ::= \quad \text{end} \quad | \quad \mu\mathbf{t}.\mathbb{T} \quad | \quad \mathbf{t} \quad | \quad \bigoplus_{i\in I} \mathrm{p}!\ell_i(S_i).\mathbb{T}_i \quad | \quad \&_{i\in I} \mathrm{p}?\ell_i(S_i).$$

```
Inductive local  : Type ≜
  | l_var : fin → local
  | l_end : local
  | l_send: part →
            list (option(sort*local)) → local
  | l_recv: part →
            list (option(sort*local)) → local
  | l_rec : local → local.
```

Local Type Trees

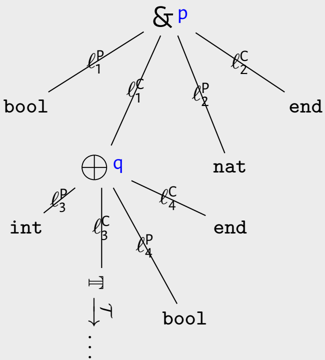
Local type trees are coinductively generated by the following grammar.

$$\mathsf{T} \quad ::= \quad \text{end} \quad | \quad \bigoplus_{i\in I} \mathrm{p}!\ell_i(S_i).\mathsf{T}_i \quad \&_{i\in I} \mathrm{p}?\ell_i(S_i).\mathsf{T}_i$$

```
CoInductive ltt: Type ≜
  | ltt_end : ltt
  | ltt_recv: part → list (option(sort*ltt)) → ltt
  | ltt_send: part → list (option(sort*ltt)) → ltt.
```

Example (Local Types $\longrightarrow$ Local Type Trees)

$$\mathbb{T} = \mu \mathbf{t}.\ \&\ \mathsf{p}? \begin{cases} \ell_1(\texttt{bool}).\ \bigoplus \mathsf{q}! \begin{cases} \ell_3(\texttt{int}).\mathbf{t} \\ \ell_4(\texttt{bool}).\texttt{end} \end{cases} \\ \ell_2(\texttt{nat}).\texttt{end} \end{cases} \xrightarrow{\mathcal{T}}$$



Projection

$$\dfrac{\forall i \in I, \mathsf{G}_i \restriction_{\mathsf{r}} \mathsf{T}_i}{\mathsf{r} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i \in I} \restriction_{\mathsf{r}} \bigoplus_{i \in I} \mathsf{q}!\ell_i(\mathsf{S}_i).\mathsf{T}_i}\ \text{[proj-send]} \qquad \dfrac{\forall i \in I, \mathsf{G}_i \restriction_{\mathsf{r}} \mathsf{T}_i}{\mathsf{p} \to \mathsf{r} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i \in I} \restriction_{\mathsf{r}} \&_{i \in I} \mathsf{q}?\ell_i(\mathsf{S}_i).\mathsf{T}_i}\ \text{[proj-recv]}$$

$$\dfrac{\mathsf{r} \notin \{\mathsf{p}, \mathsf{q}\} \quad \forall i \in I, \mathsf{r} \in \mathtt{pt}(\mathsf{G}_i) \quad \mathsf{G}_i \restriction_{\mathsf{r}} \mathsf{T}}{\mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i \in I} \restriction_{\mathsf{r}} \mathsf{T}}\ \text{[proj-cont]} \qquad \dfrac{\mathsf{r} \notin \mathtt{pt}(\mathsf{G})}{\mathsf{G} \restriction_{\mathsf{r}} \mathbf{end}}\ \text{[proj-end]}$$

Projection

$$\frac{\forall i \in I, \mathsf{G}_i \restriction_{\mathsf{r}} \mathsf{T}_i}{\mathsf{r} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i\in I} \restriction_{\mathsf{r}} \bigoplus_{i\in I} \mathsf{q}!\ell_i(\mathsf{S}_i).\mathsf{T}_i}\ \text{[proj-send]} \qquad \frac{\forall i \in I, \mathsf{G}_i \restriction_{\mathsf{r}} \mathsf{T}_i}{\mathsf{p} \to \mathsf{r} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i\in I} \restriction_{\mathsf{r}} \&_{i\in I} \mathsf{q}?\ell_i(\mathsf{S}_i).\mathsf{T}_i}\ \text{[proj-recv]}$$

$$\frac{\mathsf{r} \notin \{\mathsf{p}, \mathsf{q}\} \quad \forall i \in I, \mathsf{r} \in \mathtt{pt}(\mathsf{G}_i) \quad \mathsf{G}_i \restriction_{\mathsf{r}} \mathsf{T}}{\mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{G}_i\}_{i\in I} \restriction_{\mathsf{r}} \mathsf{T}}\ \text{[proj-cont]} \qquad \frac{\mathsf{r} \notin \mathtt{pt}(\mathsf{G})}{\mathsf{G} \restriction_{\mathsf{r}} \mathbf{end}}\ \text{[proj-end]}$$

In Coq (Projection)

```
Variant projection (R: gtt → part → ltt → Prop): gtt → part → ltt → Prop ≜
  | ...
  | proj_cont: ∀ p q r xs ys t,  p ≠ q → q ≠ r → p ≠ r →
      (isgPartsC r (gtt_send p q xs)) →
      List.Forall2 (fun u v ⇒ (u = None ∧ v = None) ∨ (∃ s g t, u = Some(s, g) ∧ v = Some t ∧ R g r t)) xs ys →
      isMerge t ys →
      projection R (gtt_send p q xs) r t.

Definition projectionC g r t ≜ paco3 projection bot3 g r t.
```

Step Relation

$$\frac{\forall i \in I \qquad \exists k \in I, \ell = \ell_k}{(\mathsf{p} \to \mathsf{q} : \{\ell_i(S_i).G_i\}_{i \in I}) \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\ G_k} \text{[st-eq]}$$

$$\frac{\{\mathsf{r}, \mathsf{s}\} \cap \{\mathsf{p}, \mathsf{q}\} = \varnothing \qquad \forall i \in I, \{\mathsf{p}, \mathsf{q}\} \subseteq \mathtt{pt}(G_i)}{(\mathsf{r} \to \mathsf{s} : \{\ell_i(S_i).G_i\}_{i \in I}) \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\ (\mathsf{r} \to \mathsf{s} : \{\ell_i(S_i).G_i \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\}_{i \in I})} \text{[st-neq]}$$

Step Relation

$$\frac{\forall i \in I \qquad \exists k \in I, \ell = \ell_k}{(\mathsf{p} \to \mathsf{q} : \{\ell_i(S_i).G_i\}_{i\in I}) \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\ G_k} \text{[st-eq]}$$

$$\frac{\{\mathsf{r},\mathsf{s}\} \cap \{\mathsf{p},\mathsf{q}\} = \varnothing \qquad \forall i \in I, \{\mathsf{p},\mathsf{q}\} \subseteq \mathrm{pt}(G_i)}{(\mathsf{r} \to \mathsf{s} : \{\ell_i(S_i).G_i\}_{i\in I}) \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\ (\mathsf{r} \to \mathsf{s} : \{\ell_i(S_i).G_i \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\}_{i\in I})} \text{[st-neq]}$$

In Coq (Step Relation)

```
Variant gttstep (R: gtt → gtt → part → part → nat → Prop): gtt → gtt → part → part → nat → Prop ≜
| ...
| stneq: ∀ p q r s xs ys n,  p ≠ q → r ≠ s → r ≠ p → r ≠ q → s ≠ p → s ≠ q →
   Forall (fun u => u = None \/ (exists s g, u = Some(s, g) /\ isgPartsC p g /\ isgPartsC q g)) xs  →
   Forall2 (fun u v ⇒ (u = None ∧ v = None) ∨ (∃ s g g', u = Some(s, g) ∧ v = Some(s, g') ∧ R g g' p q n)) xs ys →
   gttstep R (gtt_send r s xs) (gtt_send r s ys) p q n.

Definition gttstepC g1 g2 p q n ≜ paco5 gttstep bot5 g1 g2 p q n.
```

Subtyping

$$\frac{\forall i \in I, \quad \mathsf{S}_i \leq: \mathsf{S}'_i \quad \mathsf{T}_i \leqslant \mathsf{T}'_i}{\bigoplus_{i\in I} \mathsf{p}!\ell_i(\mathsf{S}_i).\mathsf{T}_i \leqslant \bigoplus_{i\in I\cup J} \mathsf{p}!\ell_i(\mathsf{S}'_i).\mathsf{T}'_i}\ \text{[sub-out]} \qquad \frac{}{\mathbf{end} \leqslant \mathbf{end}}\ \text{[sub-end]}$$

$$\frac{\forall i \in I, \quad \mathsf{S}'_i \leq: \mathsf{S}_i \quad \mathsf{T}_i \leqslant \mathsf{T}'_i}{\&_{i\in I\cup J} \mathsf{p}?\ell_i(\mathsf{S}_i).\mathsf{T}_i \leqslant \&_{i\in I} \mathsf{p}?\ell_i(\mathsf{S}'_i).\mathsf{T}'_i}\ \text{[sub-in]}$$

Subtyping

$$\dfrac{\forall i \in I, \quad S_i \leq: S'_i \quad T_i \leqslant T'_i}{\bigoplus_{i\in I} \mathrm{p}!\ell_i(S_i).T_i \leqslant \bigoplus_{i\in I\cup J} \mathrm{p}!\ell_i(S'_i).T'_i} \text{[sub-out]} \qquad \dfrac{}{\mathbf{end} \leqslant \mathbf{end}} \text{[sub-end]}$$

$$\dfrac{\forall i \in I, \quad S'_i \leq: S_i \quad T_i \leqslant T'_i}{\&_{i\in I\cup J} \mathrm{p}?\ell_i(S_i).T_i \leqslant \&_{i\in I} \mathrm{p}?\ell_i(S'_i).T'_i} \text{[sub-in]}$$

In Coq (Subtyping)

```
Fixpoint wfrec (R1: sort → sort → Prop) (R2: ltt → ltt → Prop) (l1 l2: list (option(sort*ltt))): Prop ≜
  match (l1,l2) with
    | (Datatypes.None::xs, Datatypes.None::ys)              ⇒ wfrec R1 R2 xs ys
    | (Datatypes.Some (s',t')::xs, Datatypes.Some (s,t)::ys) ⇒ R1 s' s ∧ R2 t t' ∧ wfrec R1 R2 xs ys
    | (Datatypes.None::xs, Datatypes.Some(s,t)::ys)         ⇒ wfrec R1 R2 xs ys
    | (nil, _)                                               ⇒ True
    | _                                                      ⇒ False
  end.
Inductive subtype (R: ltt → ltt → Prop): ltt → ltt → Prop ≜
  | sub_end: subtype R ltt_end ltt_end
  | sub_in : ∀ p xs ys, wfrec subsort R ys xs  → subtype R (ltt_recv p xs) (ltt_recv p ys)
  | sub_out: ∀ p xs ys, wfsend subsort R xs ys → subtype R (ltt_send p xs) (ltt_send p ys).
Definition subtypeC l1 l2 ≜ paco2 subtype bot2 l1 l2.
```

Outline

Let's now... (Subtyping is transitive)

```
Lemma stTrans: ∀ l1 l2 l3, subtypeC l1 l2 → subtypeC l2 l3 → subtypeC l1 l3.
```

Outline

Balanced Global Types

G is balanced if, for every subtree G' of G,

- if $\mathsf{p} \in \mathtt{pt}(\mathsf{G}')$, $\exists k \in \mathbb{N}$ such that
  1. For all paths γ length k from the root of G', $\mathsf{p} \in \gamma$.
  2. For all paths γ from the root of G' that end at a leaf (end), $\mathsf{p} \in \gamma$.

Balanced Global Types

G is balanced if, for every subtree G' of G,

- if $\mathsf{p} \in \mathtt{pt}(\mathsf{G}')$, $\exists k \in \mathbb{N}$ such that
  1. For all paths γ length k from the root of G', $\mathsf{p} \in \gamma$.
  2. For all paths γ from the root of G' that end at a leaf (end), $\mathsf{p} \in \gamma$.

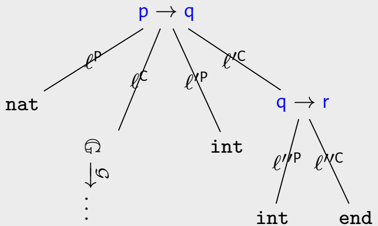
Example (Unbalanced Global Tree)

Consider the tree of the type:
$\mathbb{G} = \mu\mathbf{t}.\mathsf{p} \to \mathsf{q} : \{\ell(\mathtt{nat}).\mathbf{t}, \ell'(\mathtt{int}).\mathsf{q} \to \mathsf{r} : \{\ell''(\mathtt{int}).\mathtt{end}\}\}$
That is, a tree G such that $\mathbb{G} \xrightarrow{\mathcal{G}} \mathsf{G}$.

Example (Unbalanced Global Tree)

Consider the tree of the type:

$$\mu\mathbf{t}.\mathsf{p} \to \mathsf{q} : \{\ell(\mathtt{nat}).\mathbf{t}, \ell'(\mathtt{int}).\mathsf{q} \to \mathsf{r} : \{\ell''(\mathtt{int}).\mathtt{end}\}\} \quad \xrightarrow{\mathcal{G}}$$



Theorem

$\forall$ balanced G and p, if $p \in \mathtt{pt}(G)$ then

- $\exists\, \Gamma_G$ and L such that
  1. $G = \Gamma_G[L]$ and $p \notin \mathtt{pt}(\Gamma_G)$
  2. Each element that fills a hole in Γ_G from the list L is of one of the following forms:

     $p \to q : \{\ell_i(S_i).g_i\}_{i \in I} \quad q \to p : \{\ell_i(S_i).g_i\}_{i \in I} \quad \mathbf{end}$

  for some labels ℓ_i, sorts S_i, continuations g_i and participants q

Process Language and Sessions

$$\mathsf{P} \quad ::= \quad \mathsf{p}!\ell(e).\mathsf{P} \quad | \quad \sum_{i\in I} \mathsf{p}?\ell_i(x_i).\mathsf{P}_i \quad | \quad \text{if } e \text{ then } \mathsf{P} \text{ else } \mathsf{P} \quad | \quad \mu\mathbf{X}.\mathsf{P} \quad | \quad \mathbf{X} \quad | \quad \mathbf{0}$$

$$\mathcal{M} \quad ::= \quad \mathsf{p} \triangleleft \mathsf{P} \quad | \quad \mathcal{M} \mid \mathcal{M}$$

Type Checking

$$\frac{}{\Gamma \vdash_{\mathsf{P}} \mathbf{0} : \mathtt{end}}\,[\text{tend}] \qquad \frac{}{\Gamma, \mathbf{X} : \mathsf{T} \vdash_{\mathsf{P}} \mathbf{X} : \mathsf{T}}\,[\text{tvar}] \qquad \frac{\Gamma, \mathbf{X} : \mathsf{T} \vdash_{\mathsf{P}} \mathsf{P} : \mathsf{T}}{\Gamma \vdash_{\mathsf{P}} \mu\mathbf{X}.\mathsf{P} : \mathsf{T}}\,[\text{trec}]$$

$$\frac{\Gamma \vdash_{\mathsf{P}} \mathsf{P} : \mathsf{T} \quad \mathsf{T} \leqslant \mathsf{T}'}{\Gamma \vdash_{\mathsf{P}} \mathsf{P} : \mathsf{T}'}\,[\text{tsub}] \qquad \frac{\forall i \in I, \quad \Gamma, x_i : \mathsf{S}_i \vdash_{\mathsf{P}} \mathsf{P}_i : \mathsf{T}_i}{\Gamma \vdash_{\mathsf{P}} \sum_{i \in I} \mathsf{p}?\ell_i(x_i).\mathsf{P}_i : \ \&_{i \in I} \mathsf{p}?\ell_i(\mathsf{S}_i).\mathsf{T}_i}\,[\text{tin}] \qquad \frac{\Gamma \vdash_{\mathsf{S}} e : \mathsf{S} \quad \Gamma \vdash_{\mathsf{P}} \mathsf{P} : \mathsf{T}}{\Gamma \vdash_{\mathsf{P}} \mathsf{p}!\ell(e).\mathsf{P} : \ \bigoplus \mathsf{p}?\ell(\mathsf{S}).\mathsf{T}}\,[\text{tout}]$$

$$\frac{\Gamma \vdash_{\mathsf{S}} e : \mathtt{bool} \quad \Gamma \vdash_{\mathsf{P}} \mathsf{P}_1 : \mathsf{T} \quad \Gamma \vdash_{\mathsf{P}} \mathsf{P}_2 : \mathsf{T}}{\Gamma \vdash_{\mathsf{P}} \text{if } e \text{ then } \mathsf{P}_1 \text{ else } \mathsf{P}_2 : \mathsf{T}}\,[\text{tite}]$$

$$\frac{\forall i \in I, \quad \mathsf{G}\upharpoonright_{\mathsf{p}_i} \mathsf{T}_i \quad \vdash_{\mathsf{P}} \mathsf{P}_i : \mathsf{T}_i \quad \mathtt{pt}(\mathsf{G}) \subseteq \{\mathsf{p}_i \mid i \in I\}}{\vdash_{\mathcal{M}} \prod_{i \in I} \mathsf{p}_i \triangleleft \mathsf{P}_i : \mathsf{G}}\,[\text{tsess}]$$

Example (Aside on Proofs in with Hypothesis)

Often, we want to prove statements like:

$$\forall n \in \mathbb{N}.\, P(n)$$

which can be rewritten as

$$\forall n \in \mathbb{N}.\, H(n) \to C(n)$$

Example (Aside on Proofs in with Hypothesis)

Often, we want to prove statements like:

$$\forall n \in \mathbb{N}.\, P(n)$$

which can be rewritten as

$$\forall n \in \mathbb{N}.\, H(n) \to C(n)$$

Using induction on n, we can prove this by showing

- $H(0) \to C(0)$
- $(H(n) \to C(n)) \to (H(n+1) \to C(n+1))$

Example (Aside on Proofs in with Hypothesis)

Often, we want to prove statements like:

$$\forall n \in \mathbb{N}.\, P(n)$$

which can be rewritten as

$$\forall n \in \mathbb{N}.\, H(n) \rightarrow C(n)$$

Using induction on n, we can prove this by showing

- $H(0) \rightarrow C(0)$
- $H(n+1) \rightarrow H(n)$
- $C(n) \rightarrow C(n+1)$

Example (Aside on Proofs in with Hypothesis)

Often, we want to prove statements like:

$$\forall n \in \mathbb{N}.\, P(n)$$

which can be rewritten as

$$\forall n \in \mathbb{N}.\, H(n) \to C(n)$$

Using induction on n, we can prove this by showing

- $H(0) \to C(0)$
- $H(n+1) \to H(n)$
- $C(n) \to C(n+1)$

So often times we need to strengthen / weaken the statement to ensure we can deduce each of the implications, even if the original statement is true.

Theorem

If $\Gamma \vdash_{\mathsf{P}} \mu\mathbf{X}.\mathsf{P} : \mathsf{T}$ then $\Gamma \vdash_{\mathsf{P}} \mathsf{P}\{\mu\mathbf{X}.\mathsf{P}/\mathbf{X}\} : \mathsf{T}$

Theorem

If $\Gamma \vdash_P \mu\mathbf{X}.P : T$ then $\Gamma \vdash_P P\{\mu\mathbf{X}.P/\mathbf{X}\} : T$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : T \vdash_P P : T'$ and $\Gamma \vdash_P Q : T$, then $\Gamma \vdash_P P\{Q/\mathbf{X}\} : T'$

Theorem

If $\Gamma \vdash_P \mu\mathbf{X}.P : T$ then $\Gamma \vdash_P P\{\mu\mathbf{X}.P/\mathbf{X}\} : T$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : T \vdash_P P : T'$ and $\Gamma \vdash_P Q : T$, then $\Gamma \vdash_P P\{Q/\mathbf{X}\} : T'$

In Coq (De bruijn Indices)

- Notation where terms are invariant under α-conversion

Theorem

If $\Gamma \vdash_P \mu\mathbf{X}.P : T$ then $\Gamma \vdash_P P\{\mu\mathbf{X}.P/\mathbf{X}\} : T$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : T \vdash_P P : T'$ and $\Gamma \vdash_P Q : T$, then $\Gamma \vdash_P P\{Q/\mathbf{X}\} : T'$

In Coq (De bruijn Indices)

- Notation where terms are invariant under α-conversion
- Use natural numbers that represent the number of binders in scope between occurence to its corresponding binding μ and Σ term

Theorem

If $\Gamma \vdash_{P} \mu \mathbf{X}.P : T$ then $\Gamma \vdash_{P} P\{\mu \mathbf{X}.P/\mathbf{X}\} : T$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : T \vdash_{P} P : T'$ and $\Gamma \vdash_{P} Q : T$, then $\Gamma \vdash_{P} P\{Q/\mathbf{X}\} : T'$

In Coq (De bruijn Indices)

- Notation where terms are invariant under α-conversion
- Use natural numbers that represent the number of binders in scope between occurence to its corresponding binding μ and Σ term

Example (In the context of Lambda Calculus)

- $\lambda x.\lambda y.x$ (K combinator)

Theorem

If $\Gamma \vdash_P \mu \mathbf{X}.P : T$ then $\Gamma \vdash_P P\{\mu \mathbf{X}.P/\mathbf{X}\} : T$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : T \vdash_P P : T'$ and $\Gamma \vdash_P Q : T$, then $\Gamma \vdash_P P\{Q/\mathbf{X}\} : T'$

In Coq (De bruijn Indices)

- Notation where terms are invariant under α-conversion
- Use natural numbers that represent the number of binders in scope between occurence to its corresponding binding μ and Σ term

Example (In the context of Lambda Calculus)

- $\lambda x.\lambda y.x$ (K combinator) $\longrightarrow \lambda(\lambda\ 1)$

Theorem

If $\Gamma \vdash_{\mathsf{P}} \mu\mathbf{X}.\mathsf{P} : \mathsf{T}$ then $\Gamma \vdash_{\mathsf{P}} \mathsf{P}\{\mu\mathbf{X}.\mathsf{P}/\mathbf{X}\} : \mathsf{T}$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : \mathsf{T} \vdash_{\mathsf{P}} \mathsf{P} : \mathsf{T}'$ and $\Gamma \vdash_{\mathsf{P}} \mathsf{Q} : \mathsf{T}$, then $\Gamma \vdash_{\mathsf{P}} \mathsf{P}\{\mathsf{Q}/\mathbf{X}\} : \mathsf{T}'$

In Coq (De bruijn Indices)

- Notation where terms are invariant under α-conversion
- Use natural numbers that represent the number of binders in scope between occurence to its corresponding binding μ and Σ term

Example (In the context of Lambda Calculus)

- $\lambda x.\lambda y.x$ (K combinator) $\rightarrow \lambda(\lambda\ 1)$
- $\lambda x.\lambda y.\lambda z.x\,z\,(y\,z)$ (S combinator)

Theorem

If $\Gamma \vdash_P \mu\mathbf{X}.P : T$ then $\Gamma \vdash_P P\{\mu\mathbf{X}.P/\mathbf{X}\} : T$

Lemma

(Substitution Lemma for Process Variables) : If $\Gamma, \mathbf{X} : T \vdash_P P : T'$ and $\Gamma \vdash_P Q : T$, then $\Gamma \vdash_P P\{Q/\mathbf{X}\} : T'$

In Coq (De bruijn Indices)

- Notation where terms are invariant under α-conversion
- Use natural numbers that represent the number of binders in scope between occurence to its corresponding binding μ and Σ term

Example (In the context of Lambda Calculus)

- $\lambda x.\lambda y.x$ (K combinator) $\rightarrow \lambda(\lambda\ 1)$
- $\lambda x.\lambda y.\lambda z.x\,z\,(y\,z)$ (S combinator) $\rightarrow \lambda(\lambda(\lambda\ 2\ 0\ (1\ 0)))$

Type Checking

$$\frac{}{\Gamma_p, \Gamma_e \vdash_P \mathbf{0} : \mathtt{end}}\,[\text{tend}] \qquad \frac{}{\Gamma_p, \Gamma_e \vdash_P \mathbf{N} : (\Gamma_p)_{\mathbf{N}}}\,[\text{tvar}] \qquad \frac{(\mathsf{T} :: \Gamma_p), \Gamma_e \vdash_P \mathsf{P} : \mathsf{T}}{\Gamma_p, \Gamma_e \vdash_P \mu.\mathsf{P} : \mathsf{T}}\,[\text{trec}]$$

$$\frac{\Gamma_p, \Gamma_e \vdash_P \mathsf{P} : \mathsf{T} \quad \mathsf{T} \leqslant \mathsf{T}'}{\Gamma_p, \Gamma_e \vdash_P \mathsf{P} : \mathsf{T}'}\,[\text{tsub}] \qquad \frac{\forall i \in I, \quad \Gamma_p, (\mathsf{S}_i :: \Gamma_e) \vdash_P \mathsf{P}_i : \mathsf{T}_i}{\Gamma \vdash_P \sum_{i \in I} \mathsf{p}?\ell_i.\mathsf{P}_i : \&_{i \in I}\, \mathsf{p}?\ell_i(\mathsf{S}_i).\mathsf{T}_i}\,[\text{tin}]$$

$$\frac{\Gamma_e \vdash_S e : \mathsf{S} \quad \Gamma_p, \Gamma_e \vdash_P \mathsf{P} : \mathsf{T}}{\Gamma_p, \Gamma_e \vdash_P \mathsf{p}!\ell(e).\mathsf{P} : \bigoplus \mathsf{p}?\ell(\mathsf{S}).\mathsf{T}}\,[\text{tout}] \qquad \frac{\Gamma_e \vdash_S e : \mathtt{bool} \quad \Gamma_p, \Gamma_e \vdash_P \mathsf{P}_1 : \mathsf{T} \quad \Gamma_p, \Gamma_e \vdash_P \mathsf{P}_2 : \mathsf{T}}{\Gamma_p, \Gamma_e \vdash_P \text{if } e \text{ then } \mathsf{P}_1 \text{ else } \mathsf{P}_2 : \mathsf{T}}\,[\text{tite}]$$

Type Checking

$$\frac{}{\Gamma_p, \Gamma_e \vdash_P \mathbf{0} : \mathtt{end}}\,[\text{tend}] \qquad \frac{}{\Gamma_p, \Gamma_e \vdash_P \mathbf{N} : (\Gamma_p)_{\mathbf{N}}}\,[\text{tvar}] \qquad \frac{(T :: \Gamma_p), \Gamma_e \vdash_P P : T}{\Gamma_p, \Gamma_e \vdash_P \mu.P : T}\,[\text{trec}]$$

$$\frac{\Gamma_p, \Gamma_e \vdash_P P : T \quad T \leqslant T'}{\Gamma_p, \Gamma_e \vdash_P P : T'}\,[\text{tsub}] \qquad \frac{\forall i \in I, \quad \Gamma_p, (S_i :: \Gamma_e) \vdash_P P_i : T_i}{\Gamma \vdash_P \sum_{i \in I} p?\ell_i.P_i : \&_{i \in I}\, p?\ell_i(S_i).T_i}\,[\text{tin}]$$

$$\frac{\Gamma_e \vdash_S e : S \quad \Gamma_p, \Gamma_e \vdash_P P : T}{\Gamma_p, \Gamma_e \vdash_P p!\ell(e).P : \bigoplus p?\ell(S).T}\,[\text{tout}] \qquad \frac{\Gamma_e \vdash_S e : \mathtt{bool} \quad \Gamma_p, \Gamma_e \vdash_P P_1 : T \quad \Gamma_p, \Gamma_e \vdash_P P_2 : T}{\Gamma_p, \Gamma_e \vdash_P \text{if } e \text{ then } P_1 \text{ else } P_2 : T}\,[\text{tite}]$$

Lemma

(Substitution Lemma for Process Variables with De Bruijn Indices) : If

- $(\Gamma_{p_1} +\!\!\!+ (T :: \Gamma_{p_2})), \Gamma_e \vdash_P P : T'$
- $(\Gamma_{p_1} +\!\!\!+ \Gamma_{p_2}), \Gamma_e \vdash_P Q_{\{m,n\}} : T$

then $(\Gamma_{p_1} +\!\!\!+ \Gamma_{p_2}), \Gamma_e \vdash_P P\{Q/\,||\Gamma_{p_1}||\}_{\{m,n\}} : T'$

Lemma

$\forall$ balanced G and ℓ, if $G \upharpoonright_p \bigoplus_{i\in I} q!\ell_i(S_i).T_i$, $G \upharpoonright_q \&_{j\in J} q?\ell'_j(S'_j).T'_j$, and $\exists\, k_i \in I, k_j \in J$ such that $\ell = \ell_{k_i}$ and $\ell = \ell'_{k_j}$, then $\exists\, \Gamma_G$ and L such that

- $G = \Gamma_G[L]$
- $p, q \notin pt(\Gamma_G)$

Moreover, There is some K such that,

- $\forall G' \in L, G' = p \to q : \{\ell''_k(S''_k).g_{G'k}\}_{k\in K}$ for some $\{g_{G'k}\}_{G'\in L\ k\in K}$
- $\exists k \in K$ such that
  1. $\ell = \ell''_k$
  2. $S''_k = S_{k_i} = S'_{k_j}$

Theorem

$\forall$ balanced G and ℓ, if $\mathsf{G} \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\ \mathsf{G}'$, $\mathsf{G} \upharpoonright_{\mathsf{p}} \bigoplus_{i \in I} \mathsf{q}!\ell_i(\mathsf{S}_i).\mathsf{T}_i$, $\mathsf{G} \upharpoonright_{\mathsf{q}} \&_{j \in J} \mathsf{q}?\ell'_j(\mathsf{S}'_j).\mathsf{T}'_j$, and $\exists\, k_i \in I, k_j \in J$ such that $\ell = \ell_{k_i}$ and $\ell = \ell'_{k_j}$, then

- $\mathsf{G}' \upharpoonright_{\mathsf{p}} \mathsf{T}_{k_i}$
- $\mathsf{G}' \upharpoonright_{\mathsf{q}} \mathsf{T}'_{k_j}$
- if $\mathsf{G} \upharpoonright_{\mathsf{r}} \mathsf{T}$, then $\mathsf{G}' \upharpoonright_{\mathsf{r}} \mathsf{T}$

Theorem

$\forall$ balanced G and ℓ, if $\mathsf{G} \setminus \mathsf{p} \xrightarrow{\ell} \mathsf{q}\ \mathsf{G}'$, $\mathsf{G} \upharpoonright_{\mathsf{p}} \bigoplus_{i \in I} \mathsf{q}!\ell_i(\mathsf{S}_i).\mathsf{T}_i$, $\mathsf{G} \upharpoonright_{\mathsf{q}} \&_{j \in J} \mathsf{q}?\ell'_j(\mathsf{S}'_j).\mathsf{T}'_j$, and $\exists\, k_i \in I, k_j \in J$ such that $\ell = \ell_{k_i}$ and $\ell = \ell'_{k_j}$, then

- $\mathsf{G}' \upharpoonright_{\mathsf{p}} \mathsf{T}_{k_i}$
- $\mathsf{G}' \upharpoonright_{\mathsf{q}} \mathsf{T}'_{k_j}$
- if $\mathsf{G} \upharpoonright_{\mathsf{r}} \mathsf{T}$, then $\mathsf{G}' \upharpoonright_{\mathsf{r}} \mathsf{T}$

In Coq (First Bullet)

```
Lemma _3_19_1_helper : ∀ p q l G G' LP LQ S T S' T',
    wfgC G →
    projectionC G p (ltt_send q LP) →
    onth l LP = Some(S, T) →
    projectionC G q (ltt_recv p LQ) →
    onth l LQ = Some(S', T') →
    gttstepC G G' p q l →
    projectionC G' p T.
Proof.
    ...
Qed.
```

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i \in I}$
- $G' = g_\ell$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i\in I}$
- $G' = g_\ell$
- $G \upharpoonright_p \bigoplus_{i\in I} q!\ell_i(S_i).T_i$, where $\forall i \in I$, $g_i \upharpoonright_p T_i$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_{G} for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_{\mathsf{G}} = [\,]_k$

- $\Gamma_{\mathsf{G}}[\mathsf{L}] = \mathsf{G} = \mathsf{p} \to \mathsf{q} : \{\ell_i(\mathsf{S}_i).\mathsf{g}_i\}_{i \in I}$
- $\mathsf{G}' = \mathsf{g}_\ell$
- $\mathsf{G} \upharpoonright_{\mathsf{p}} \bigoplus_{i \in I} \mathsf{q}!\ell_i(\mathsf{S}_i).\mathsf{T}_i$, where $\forall i \in I$, $\mathsf{g}_i \upharpoonright_{\mathsf{p}} \mathsf{T}_i$
- $\mathsf{G}' \upharpoonright_{\mathsf{p}} \mathsf{T}_\ell$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i\in I}$
- $G' = g_\ell$
- $G \restriction_p \bigoplus_{i\in I} q!\ell_i(S_i).T_i$, where $\forall i \in I$, $g_i \restriction_p T_i$
- $G' \restriction_p T_\ell$

Inductive case : $\Gamma_G = s \to s' : \{\ell_i(S_i).\Gamma_{G_i}\}_{i\in I}$, $\{s, s'\} \cap \{p, q\} = \varnothing$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i \in I}$
- $G' = g_\ell$
- $G \upharpoonright_p \bigoplus_{i \in I} q!\ell_i(S_i).T_i$, where $\forall i \in I$, $g_i \upharpoonright_p T_i$
- $G' \upharpoonright_p T_\ell$

Inductive case : $\Gamma_G = s \to s' : \{\ell_i(S_i).\Gamma_{G_i}\}_{i \in I}$, $\{s, s'\} \cap \{p, q\} = \varnothing$

- $\Gamma_G[L] = G = s \to s' : \{\ell_i(S_i).g_i\}_{i \in I}$, where $\forall i \in I$, $\Gamma_{G_i}[L] = g_i$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i \in I}$
- $G' = g_\ell$
- $G \upharpoonright_p \bigoplus_{i \in I} q!\ell_i(S_i).T_i$, where $\forall i \in I, g_i \upharpoonright_p T_i$
- $G' \upharpoonright_p T_\ell$

Inductive case : $\Gamma_G = s \to s' : \{\ell_i(S_i).\Gamma_{G_i}\}_{i \in I}, \{s, s'\} \cap \{p, q\} = \varnothing$

- $\Gamma_G[L] = G = s \to s' : \{\ell_i(S_i).g_i\}_{i \in I}$, where $\forall i \in I, \Gamma_{G_i}[L] = g_i$
- $G' = s \to s' : \{\ell_i(S_i).g_i'\}_{i \in I}$, where $g_i \setminus p \xrightarrow{\ell} q\; g_i'$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.
Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i \in I}$
- $G' = g_\ell$
- $G \upharpoonright_p \bigoplus_{i \in I} q!\ell_i(S_i).T_i$, where $\forall i \in I$, $g_i \upharpoonright_p T_i$
- $G' \upharpoonright_p T_\ell$

Inductive case : $\Gamma_G = s \to s' : \{\ell_i(S_i).\Gamma_{G_i}\}_{i \in I}$, $\{s, s'\} \cap \{p, q\} = \varnothing$

- $\Gamma_G[L] = G = s \to s' : \{\ell_i(S_i).g_i\}_{i \in I}$, where $\forall i \in I$, $\Gamma_{G_i}[L] = g_i$
- $G' = s \to s' : \{\ell_i(S_i).g_i'\}_{i \in I}$, where $g_i \setminus p \xrightarrow{\ell} q\, g_i'$
- $G \upharpoonright_p \bigoplus_{j \in J} q!\ell_j(S_j).T_j$, where $\forall i \in I$, $g_i \upharpoonright_p \bigoplus_{j \in J} q!\ell_j(S_j).T_j^i$ and $T_j \sim T_j^i$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.

Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i \in I}$
- $G' = g_\ell$
- $G \upharpoonright_p \bigoplus_{i \in I} q!\ell_i(S_i).T_i$, where $\forall i \in I$, $g_i \upharpoonright_p T_i$
- $G' \upharpoonright_p T_\ell$

Inductive case : $\Gamma_G = s \to s' : \{\ell_i(S_i).\Gamma_{G_i}\}_{i \in I}$, $\{s, s'\} \cap \{p, q\} = \varnothing$

- $\Gamma_G[L] = G = s \to s' : \{\ell_i(S_i).g_i\}_{i \in I}$, where $\forall i \in I$, $\Gamma_{G_i}[L] = g_i$
- $G' = s \to s' : \{\ell_i(S_i).g'_i\}_{i \in I}$, where $g_i \setminus p \xrightarrow{\ell} q\, g'_i$
- $G \upharpoonright_p \bigoplus_{j \in J} q!\ell_j(S_j).T_j$, where $\forall i \in I$, $g_i \upharpoonright_p \bigoplus_{j \in J} q!\ell_j(S_j).T^i_j$ and $T_j \sim T^i_j$
- $\forall i \in I$, $g'_i \upharpoonright_p T^i_\ell$

Example (Proof Sketch of first bullet)

Idea : induction on Γ_G for a corresponding L that satisfies the previous lemma.

Base case : $\Gamma_G = [\,]_k$

- $\Gamma_G[L] = G = p \to q : \{\ell_i(S_i).g_i\}_{i \in I}$
- $G' = g_\ell$
- $G \upharpoonright_p \bigoplus_{i \in I} q!\ell_i(S_i).T_i$, where $\forall i \in I$, $g_i \upharpoonright_p T_i$
- $G' \upharpoonright_p T_\ell$

Inductive case : $\Gamma_G = s \to s' : \{\ell_i(S_i).\Gamma_{G_i}\}_{i \in I}, \{s, s'\} \cap \{p, q\} = \varnothing$

- $\Gamma_G[L] = G = s \to s' : \{\ell_i(S_i).g_i\}_{i \in I}$, where $\forall i \in I$, $\Gamma_{G_i}[L] = g_i$
- $G' = s \to s' : \{\ell_i(S_i).g_i'\}_{i \in I}$, where $g_i \setminus p \xrightarrow{\ell} q\, g_i'$
- $G \upharpoonright_p \bigoplus_{j \in J} q!\ell_j(S_j).T_j$, where $\forall i \in I$, $g_i \upharpoonright_p \bigoplus_{j \in J} q!\ell_j(S_j).T_j^i$ and $T_j \sim T_j^i$
- $\forall i \in I$, $g_i' \upharpoonright_p T_\ell^i$
- $G' \upharpoonright_p T_\ell$

Theorem

If $\vdash \mathcal{M} : \mathsf{G}$ and $\mathcal{M} \rightarrow_\beta \mathcal{M}'$, then

- $\exists \mathsf{G}'$ such that $\vdash \mathcal{M}' : \mathsf{G}'$ and $\mathsf{G} \rightarrow \mathsf{G}'$

Theorem

If $\vdash \mathcal{M} : \mathsf{G}$ and $\mathcal{M} \rightarrow_\beta \mathcal{M}'$, then

- $\exists \mathsf{G}'$ such that $\vdash \mathcal{M}' : \mathsf{G}'$ and $\mathsf{G} \rightarrow \mathsf{G}'$

Session Step

$$\frac{j \in I \quad e \downarrow v}{\mathsf{p} \triangleleft \sum_{i \in I} \mathsf{q}?\ell_i(x_i).\mathsf{P}_i \mid \mathsf{q} \triangleleft \mathsf{p}!\ell_j(e).\mathsf{Q} \mid \prod_{i \in J} M_i \rightarrow_\beta \mathsf{p} \triangleleft \mathsf{P}_j\{v/x_j\} \mid \mathsf{q} \triangleleft \mathsf{Q} \mid \prod_{i \in J} M_i} \text{[R-COMM]}$$

$$\frac{e \downarrow \mathsf{true}}{\mathsf{p} \triangleleft \mathsf{if}\ e\ \mathsf{then}\ \mathsf{P}\ \mathsf{else}\ \mathsf{Q} \mid \prod_{i \in I} M_i \rightarrow_\beta \mathsf{p} \triangleleft P \mid \prod_{i \in I} M_i} \text{[T-COND]}$$

$$\frac{e \downarrow \mathsf{false}}{\mathsf{p} \triangleleft \mathsf{if}\ e\ \mathsf{then}\ \mathsf{P}\ \mathsf{else}\ \mathsf{Q} \mid \prod_{i \in I} M_i \rightarrow_\beta \mathsf{p} \triangleleft \mathsf{Q} \mid \prod_{i \in I} M_i} \text{[F-COND]} \qquad \frac{M_1' \Rrightarrow M_1 \quad M_1 \rightarrow_\beta M_2 \quad M_2 \Rrightarrow M_2'}{M_1' \rightarrow_\beta M_2'} \text{[R-STRUCT]}$$

Theorem

If

- $\vdash \mathcal{M} : \mathsf{G}$
- $\mathcal{M} \rightarrow_{\beta}^{\star} \mathcal{M}'$

Then

- $\exists\, \mathsf{G}'$ such that $\vdash \mathcal{M}' : \mathsf{G}'$ and $\mathsf{G} \implies \mathsf{G}'$

Thanks! & Questions?