Formalizing TT-lifting in HOL-Nominal

Christian Doczkal

Advisor: Dr. Jan Schwinghammer
Supervisor: Prof. Gert Smolka

May 15, 2009

Moggi's computational metalanguage

- terms and types:

$$\tau ::= b \mid \tau \to \tau \mid T\ \tau$$
$$t ::= x \mid \lambda x.t \mid t\ t \mid [t] \mid t\ \texttt{to}\ x\ \texttt{in}\ t$$

- typing rules:

$$\frac{\Gamma \vdash t : \tau}{\Gamma \vdash [t] : T\,\tau} \qquad \frac{\Gamma \vdash s : T\,\sigma \qquad \Gamma, x : \sigma \vdash t : T\,\tau}{\Gamma \vdash s\ \texttt{to}\ x\ \texttt{in}\ t : T\,\tau}$$

- reductions:

$$\begin{array}{ll} T.\beta & [s]\ \texttt{to}\ x\ \texttt{in}\ t \mapsto t[x ::= s] \\ T.\eta & s\ \texttt{to}\ x\ \texttt{in}\ [x] \mapsto s \\ T.assoc & (s\ \texttt{to}\ x\ \texttt{in}\ t)\ \texttt{to}\ y\ \texttt{in}\ u \mapsto s\ \texttt{to}\ x\ \texttt{in}\ (t\ \texttt{to}\ y\ \texttt{in}\ u) \end{array}$$

Main Goal

Strong Normalization Theorem (Isar Version)

theorem typing-implies-SN:
assumes a: $\Gamma \vdash t : \tau$
shows SN(t)

- Originally shown by Benton et al. [1998]
- I formalized the proof of Lindley and Stark [2005]

The Girard-Tait approach

1. Define a type indexed family of relations red_τ
2. Show by induction on the type structure

$$\text{(CR1)} \qquad t \in red_\tau \Rightarrow SN(t)$$

$$\text{(CR2)} \qquad t \in red_\tau \wedge t \mapsto t' \Rightarrow t' \in red_\tau$$

$$\text{(CR3)} \qquad neutral(t) \wedge (\forall t'.t \mapsto t' \Rightarrow t' \in red_\tau) \Rightarrow t \in red_\tau$$

3. Prove $\Gamma \vdash t : \tau \Rightarrow t \in red_\tau$ by induction on the typing derivation

First attempt at defining reducibility:

$$t \in red_b \equiv SN\ t$$

$$t \in red_{\sigma \to \tau} \equiv \forall u \in red_\sigma . t\ u \in red_\tau$$

$$t \in red_{T\,\sigma} \equiv \forall u \in X . t\ \texttt{to}\ x\ \texttt{in}\ u \in X$$

The Girard-Tait approach

1. Define a type indexed family of relations red_τ
2. Show by induction on the type structure

$$\text{(CR1)} \qquad t \in red_\tau \Rightarrow SN(t)$$

$$\text{(CR2)} \qquad t \in red_\tau \wedge t \mapsto t' \Rightarrow t' \in red_\tau$$

$$\text{(CR3)} \qquad neutral(t) \wedge (\forall t'.t \mapsto t' \Rightarrow t' \in red_\tau) \Rightarrow t \in red_\tau$$

3. Prove $\Gamma \vdash t : \tau \Rightarrow t \in red_\tau$ by induction on the typing derivation

First attempt at defining reducibility:

$$t \in red_b \equiv SN\ t$$

$$t \in red_{\sigma \to \tau} \equiv \forall u \in red_\sigma . t\ u \in red_\tau$$

$$t \in red_{T\,\sigma} \equiv \forall u \in X . t\ \texttt{to}\ x\ \texttt{in}\ u \in X$$

The Girard-Tait approach

1. Define a type indexed family of relations red_τ
2. Show by induction on the type structure

(CR1) $$t \in red_\tau \Rightarrow SN(t)$$

(CR2) $$t \in red_\tau \wedge t \mapsto t' \Rightarrow t' \in red_\tau$$

(CR3) $neutral(t) \wedge (\forall t'.\ t \mapsto t' \rightarrow t' \in red_\tau) \Rightarrow t \in red_\tau$

$$\frac{\Gamma \vdash s : T\,\sigma \qquad \Gamma, x : \sigma \vdash t : T\,\tau}{\Gamma \vdash s \texttt{ to } x \texttt{ in } t : T\,\tau}$$

$$t \in red_b \equiv SN\ t$$

$$t \in red_{\sigma \to \tau} \equiv \forall u \in red_\sigma . t\ u \in red_\tau$$

$$t \in red_{T\,\sigma} \equiv \forall u \in X . t \texttt{ to } x \texttt{ in } u \in X$$

$\top\top$-lifting

Definition (stack)

$$K ::= Id \mid [y]n \gg L$$

$$t \star Id = t$$
$$t \star ([y]n \gg L) = (t \texttt{ to } y \texttt{ in } n) \star L$$

Definition (reducibility)

$$t \in red_b \equiv SN\ t$$
$$t \in red_{\sigma \to \tau} \equiv \forall u \in red_\sigma . t\ u \in red_\tau$$
$$t \in red_{T\,\sigma} \equiv \forall K \in red_\sigma^\top . SN(t \star K)$$

$$K \in red_\sigma^\top \equiv \forall s \in red_\sigma . SN([s] \star K)$$

$\top\top$-lifting

Definition (stack)

$$K ::= Id \mid [y]n \gg L$$

$$t \star Id = t$$
$$t \star ([y]n \gg L) = (t \texttt{ to } y \texttt{ in } n) \star L$$

$$t \star ([y_1]n_1 \gg [y_2]n_2 \gg [y_3]n_3 \gg Id) =$$
$$((t \texttt{ to } y_1 \texttt{ in } n_1) \texttt{ to } y_2 \texttt{ in } n_2) \texttt{ to } y_3 \texttt{ in } n_3$$

$$t \in red_{T\,\sigma} \equiv \forall K \in red_\sigma^\top . SN(t \star K)$$

$$K \in red_\sigma^\top \equiv \forall s \in red_\sigma . SN([s] \star K)$$

$\top\top$-lifting

Definition (stack)

$$K ::= Id \mid [y]n \gg L$$

$$t \star Id = t$$
$$t \star ([y]n \gg L) = (t \texttt{ to } y \texttt{ in } n) \star L$$

Definition (reducibility)

$$t \in red_b \equiv SN\ t$$
$$t \in red_{\sigma \to \tau} \equiv \forall u \in red_\sigma . t\ u \in red_\tau$$
$$t \in red_{T\,\sigma} \equiv \forall K \in red_\sigma^\top . SN(t \star K)$$

$$K \in red_\sigma^\top \equiv \forall s \in red_\sigma . SN([s] \star K)$$

Strong Normalization

For my formalization I use an inductive characterization

Definition (strong normalization)

$$SN\ t \equiv \forall t'.t \mapsto t' \Rightarrow SN\ t'$$

Definition (stack strong normalization)

$$SSN\ k \equiv \forall k'.k \mapsto_k k' \Rightarrow SSN\ k'$$

where $k \mapsto_k k' \equiv \forall t.\ t \star k \mapsto t \star k'$

Lemma (Properties of reducibility)

$$\begin{array}{ll} (CR1) & t \in red_\tau \Rightarrow SN(t) \\ (CR2) & t \in red_\tau \wedge t \mapsto t' \Rightarrow t' \in red_\tau \\ (CR3) & neutral(t) \wedge (\forall t'.\, t \mapsto t' \Rightarrow t' \in red_\tau) \Rightarrow t \in red_\tau \end{array}$$

- Proof by induction on the type structure
- Consider case CR3 for $T\,\sigma$ in detail
- First the informal proof

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED^{\top}_{\sigma}$. First, we have that $[x] \star k$ is SN, as $x \in RED_{\sigma}$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$. Application $t \star k$ may reduce as follows:

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED^{\top}_{\sigma}$ and*
- *$t \star k'$ where $k \mapsto k'$. Here we have $k' \in RED^{\top}_{\sigma}$ with $\max(k') < \max(k)$, so by induction hypothesis $t \star k'$ is SN.*

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED^{\top}_{\sigma}$, and so $t \in RED_{T\,\sigma}$ as required.

(CR3): $neutral(t) \wedge (\forall t'.t \mapsto t' \Rightarrow t' \in red_{\tau}) \Rightarrow t \in red_{\tau}$

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED^{\top}_{\sigma}$. First, we have that $[x] \star k$ is SN, as $x \in RED_{\sigma}$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$.

Application $t \star k$ may reduce as follows:

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED^{\top}_{\sigma}$ and*

(CR3): $neutral(t) \wedge (\forall t'.t \mapsto t' \Rightarrow t' \in red_{\tau}) \Rightarrow t \in red_{\tau}$

this we have $k' \in RED^{\top}_{\sigma}$ with $\max(k') < \max(k)$, so by induction hypothesis $t \star k'$ is SN.

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED^{\top}_{\sigma}$, and so $t \in RED_{T\,\sigma}$ as required.

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED_\sigma^\top$. First, we have that $[x] \star k$ is SN, as $x \in RED_\sigma$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$. *Application $t \star k$ may reduce as follows:*

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED_\sigma^\top$ and $t' \in RED_{T\,\sigma}$.*
- *$t \star k'$, where $k \mapsto k'$. For any $s \in RED_\sigma$, $[s] \star k$ is SN as $k \in RED_\sigma^\top$; and $[s] \star k \mapsto [s] \star k'$, so $[s] \star k'$ is also SN. From this we have $k' \in RED_\sigma^\top$ with $\max(k') < \max(k)$, so by induction hypothesis $t \star k'$ is SN.*

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED_\sigma^\top$, and so $t \in RED_{T\,\sigma}$ as required.

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED^{\top}_{\sigma}$. First, we have that $[x] \star k$ is SN, as $x \in RED_{\sigma}$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$. *Application $t \star k$ may reduce as follows:*

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED^{\top}_{\sigma}$ and $t' \in RED_{T\,\sigma}$.*
- *$t \star k'$, where $k \mapsto k'$. For any $s \in RED_{\sigma}$, $[s] \star k$ is SN as $k \in RED^{\top}_{\sigma}$; and $[s] \star k \mapsto [s] \star k'$, so $[s] \star k'$ is also SN. From this we have $k' \in RED^{\top}_{\sigma}$ with* $\max(k') < \max(k)$, *so by induction hypothesis $t \star k'$ is SN.*

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED^{\top}_{\sigma}$, and so $t \in RED_{T\,\sigma}$ as required.

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED_\sigma^\top$. First, we have that $[x] \star k$ is SN, as $x \in RED_\sigma$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$. Application $t \star k$ may reduce as follows:

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED_\sigma^\top$ and $t' \in RED_{T\,\sigma}$.*
- *$t \star k'$, where $k \mapsto k'$. For any $s \in RED_\sigma$, $[s] \star k$ is SN as $k \in RED_\sigma^\top$; and $[s] \star k \mapsto [s] \star k'$, so $[s] \star k'$ is also SN. From this we have $k' \in RED_\sigma^\top$ with $\max(k') < \max(k)$, so by induction hypothesis $t \star k'$ is SN.*

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED_\sigma^\top$, and so $t \in RED_{T\,\sigma}$ as required.

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED^{\top}_{\sigma}$. First, we have that $[x] \star k$ is SN, as $x \in RED_{\sigma}$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$. *Application $t \star k$ may reduce as follows:*

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED^{\top}_{\sigma}$ and $t' \in RED_{T\,\sigma}$.*
- *$t \star k'$, where $k \mapsto k'$. For any $s \in RED_{\sigma}$, $[s] \star k$ is SN as $k \in RED^{\top}_{\sigma}$; and $[s] \star k \mapsto [s] \star k'$, so $[s] \star k'$ is also SN. From this we have $k' \in RED^{\top}_{\sigma}$ with* $\max(k') < \max(k)$, *so by induction hypothesis $t \star k'$ is SN.*

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED^{\top}_{\sigma}$, and so $t \in RED_{T\,\sigma}$ as required.

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED^{\top}_{\sigma}$. First, we have that $[x] \star k$ is SN, as $x \in RED_{\sigma}$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$.

Application $t \star k$ may reduce as follows:

- *$t' \star k$, where $t \mapsto t'$, which is SN as $k \in RED^{\top}_{\sigma}$ and $t' \in RED_{T\,\sigma}$.*
- *$t \star k'$, where $k \mapsto k'$. For any $s \in RED_{\sigma}$, $[s] \star k$ is SN as $k \in RED^{\top}_{\sigma}$; and $[s] \star k \mapsto [s] \star k'$, so $[s] \star k'$ is also SN. From this we have $k' \in RED^{\top}_{\sigma}$ with* $\max(k') < \max(k)$, *so by induction hypothesis $t \star k'$ is SN.*

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED^{\top}_{\sigma}$, and so $t \in RED_{T\,\sigma}$ as required.

CR3 for $T\,\sigma$

Let t be neutral such that $t' \in RED_{T\,\sigma}$ whenever $t \mapsto t'$. We have to show that $(t \star k)$ is SN for each $k \in RED^{\top}_{\sigma}$. First, we have that $[x] \star k$ is SN, as $x \in RED_{\sigma}$ by the induction hypothesis. Hence k itself is SN , and we can work by induction on $\max(k)$.

Application $t \star k$ may reduce as follows:

- *$t' \star k$ where $t \mapsto t'$ which is SN as $k \in RED^{\top}$ and*

$$T.\beta \qquad [s]\ \texttt{to}\ x\ \texttt{in}\ t \mapsto t[x ::= s]$$
$$T.assoc \quad (s\ \texttt{to}\ x\ \texttt{in}\ t)\ \texttt{to}\ y\ \texttt{in}\ u \mapsto s\ \texttt{to}\ x\ \texttt{in}\ (t\ \texttt{to}\ y\ \texttt{in}\ u)$$

  induction hypothesis $t \star k'$ is SN.

There are no other possibilities as t is neutral. Hence $t \star k$ is strongly normalizing for every $k \in RED^{\top}_{\sigma}$, and so $t \in RED_{T\,\sigma}$ as required.

Closing the Gap

$$NEUT\ t \equiv (\exists x.\ t = Var\ x) \vee (\exists u\ v.\ t = App\ u\ v)$$

$$\frac{t \star k \mapsto r \qquad \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \qquad NEUT\ t \qquad \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P}{P}$$

- Want to prove this by induction on k
- Case $k = Id$ is easy
- Case $k = [y]n \gg l$ Want to use induction hypothesis with $t^* = t$ `to` y `in` n and $k^* = l$

Closing the Gap

$$NEUT\ t \equiv (\exists x.\ t = Var\ x) \vee (\exists u\ v.\ t = App\ u\ v)$$

$$\frac{\begin{array}{ll} t \star k \mapsto r & \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ NEUT\ t & \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \end{array}}{P}$$

- Want to prove this by induction on k
- Case $k = Id$ is easy
- Case $k = [y]n \gg l$ Want to use induction hypothesis with $t^* = t$ `to` y `in` n and $k^* = l$

Closing the Gap

$$NEUT\ t \equiv (\exists x.\ t = Var\ x) \vee (\exists u\ v.\ t = App\ u\ v)$$

$$\frac{\begin{array}{ll} t \star k \mapsto r & \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ NEUT\ t & \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \end{array}}{P}$$

- Want to prove this by induction on k
- Case $k = Id$ is easy
- Case $k = [y]n \gg l$ Want to use induction hypothesis with $t^* = t\ \texttt{to}\ y\ \texttt{in}\ n$ and $k^* = l$

Closing the Gap

$$NEUT\ t \equiv (\exists x.\ t = Var\ x) \vee (\exists u\ v.\ t = App\ u\ v)$$

$$\frac{\begin{array}{ll} t \star k \mapsto r & \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ NEUT\ t & \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \end{array}}{P}$$

- Want to prove this by induction on k
- Case $k = Id$ is easy
- Case $k = [y]n \gg l$ Want to use induction hypothesis with $t^* = t\ \texttt{to}\ y\ \texttt{in}\ n$ and $k^* = l$

Closing the Gap

$$NEUT\ t \equiv (\exists x.\ t = Var\ x) \vee (\exists u\ v.\ t = App\ u\ v)$$

$$\frac{\begin{array}{ll} t \star k \mapsto r & \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ NEUT\ t & \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \end{array}}{P}$$

- Want to prove this by induction on k
- Case $k = Id$ is easy
- Case $k = [y]n \gg l$ Want to use induction hypothesis with $t^* = t$ `to` y `in` n and $k^* = l$ *not* neutral

Closing the Gap

$$NEUT\ t \equiv (\exists x.\ t = Var\ x) \vee (\exists u\ v.\ t = App\ u\ v)$$

$$\frac{\begin{array}{ll} t \star k \mapsto r & \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ NEUT\ t & \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \end{array}}{P}$$

- Want to prove this by induction on k
- Case $k = Id$ is easy
- Case $k = [y]n \gg l$ Want to use induction hypothesis with $t^* = t$ `to` y `in` n and $k^* = l$ *not* neutral

Closing the Gap

$$\frac{\begin{array}{c} t \star k \mapsto r \qquad \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \\ \bigwedge s\ y\ n\ l\ .[\![\ y \sharp l\ ;\ y \sharp s\ ;\ t = [s]\ ;\ k = [y]n \gg l\ ; \\ r = (n[y::=s]) \star l\ ]\!] \Longrightarrow P \\ \bigwedge u\ x\ v\ y\ n\ l.[\![\ x \sharp (y,u,n)\ ;\ y \sharp (v,u)\ ;\ t = u\ to\ x\ in\ v\ ; \\ k = [y]n \gg l\ ;\ r = (u\ to\ x\ in\ (v\ to\ y\ in\ n)) \star l\ ]\!] \Longrightarrow P \end{array}}{P}$$

- Case $k = Id$ is still easy
- Case $k = [y]n \gg l$ We use the induction hypothesis with with $t^* = t\ \mathtt{to}\ y\ \mathtt{in}\ n$ and $k^* = l$
- Most cases are easy but tedious

Closing the Gap

$$T.\beta \qquad [s]\ \mathtt{to}\ y\ \mathtt{in}\ n \mapsto n[y ::= s]$$
$$T.assoc \quad (u\ \mathtt{to}\ x\ \mathtt{in}\ v)\ \mathtt{to}\ y\ \mathtt{in}\ n \mapsto u\ \mathtt{to}\ x\ \mathtt{in}\ (v\ \mathtt{to}\ y\ \mathtt{in}\ n)$$

$$\frac{\begin{array}{l}\bigwedge s\ y\ n\ l\ .[\![\ y \sharp l\ ;\ y \sharp s\ ;\ t = [s]\ ;\ k = [y]n \gg l\ ; \\ \qquad r = (n[y::=s]) \star l\ ]\!] \Longrightarrow P \\ \bigwedge u\ x\ v\ y\ n\ l.[\![\ x \sharp (y,u,n)\ ;\ y \sharp (v,u)\ ;\ t = u\ to\ x\ in\ v\ ; \\ \qquad k = [y]n \gg l\ ;\ r = (u\ to\ x\ in\ (v\ to\ y\ in\ n)) \star l\ ]\!] \Longrightarrow P\end{array}}{P}$$

- Case $k = Id$ is still easy
- Case $k = [y]n \gg l$ We use the induction hypothesis with with $t^* = t\ \mathtt{to}\ y\ \mathtt{in}\ n$ and $k^* = l$
- Most cases are easy but tedious

Closing the Gap

$$\frac{\begin{array}{c} t \star k \mapsto r \qquad \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \\ \bigwedge s\ y\ n\ l\ .[\![\ y \sharp l\ ;\ y \sharp s\ ;\ t = [s]\ ;\ k = [y]n \gg l\ ; \\ r = (n[y::=s]) \star l\ ]\!] \Longrightarrow P \\ \bigwedge u\ x\ v\ y\ n\ l.[\![\ x \sharp (y,u,n)\ ;\ y \sharp (v,u)\ ;\ t = u\ to\ x\ in\ v\ ; \\ k = [y]n \gg l\ ;\ r = (u\ to\ x\ in\ (v\ to\ y\ in\ n)) \star l\ ]\!] \Longrightarrow P \end{array}}{P}$$

- Case $k = Id$ is still easy
- Case $k = [y]n \gg l$ We use the induction hypothesis with with $t^* = t\ \mathtt{to}\ y\ \mathtt{in}\ n$ and $k^* = l$
- Most cases are easy but tedious

Closing the Gap

$$\frac{\begin{array}{c} t \star k \mapsto r \qquad \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \\ \bigwedge s\ y\ n\ l\ .[\![\ y \sharp l\ ;\ y \sharp s\ ;\ t = [s]\ ;\ k = [y]n \gg l\ ; \\ r = (n[y::=s]) \star l\ ]\!] \Longrightarrow P \\ \bigwedge u\ x\ v\ y\ n\ l.[\![\ x \sharp (y,u,n)\ ;\ y \sharp (v,u)\ ;\ t = u\ to\ x\ in\ v\ ; \\ k = [y]n \gg l\ ;\ r = (u\ to\ x\ in\ (v\ to\ y\ in\ n)) \star l\ ]\!] \Longrightarrow P \end{array}}{P}$$

- Case $k = Id$ is still easy
- Case $k = [y]n \gg l$ We use the induction hypothesis with with $t^* = t\ \mathtt{to}\ y\ \mathtt{in}\ n$ and $k^* = l$
- Most cases are easy but tedious

Closing the Gap

$$\frac{\begin{array}{c} t \star k \mapsto r \qquad \bigwedge t'.\ [\![\ t \mapsto t';\ r = t' \star k]\!] \Longrightarrow P \\ \bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k']\!] \Longrightarrow P \\ \bigwedge s\ y\ n\ l\ .[\![\ y \sharp l\ ;\ y \sharp s\ ;\ t = [s]\ ;\ k = [y]n \gg l\ ; \\ r = (n[y::=s]) \star l\ ]\!] \Longrightarrow P \\ \bigwedge u\ x\ v\ y\ n\ l.[\![\ x \sharp (y,u,n)\ ;\ y \sharp (v,u)\ ;\ t = u\ to\ x\ in\ v\ ; \\ k = [y]n \gg l\ ;\ r = (u\ to\ x\ in\ (v\ to\ y\ in\ n)) \star l\ ]\!] \Longrightarrow P \end{array}}{P}$$

- Case $k = Id$ is still easy
- Case $k = [y]n \gg l$ We use the induction hypothesis with with $t^* = t\ \texttt{to}\ y\ \texttt{in}\ n$ and $k^* = l$
- Most cases are easy but tedious

Closing the Gap

One illustrative case:

In the case $t \texttt{ to } y \texttt{ in } n \mapsto r'$ consider the case $n \mapsto n'$

- For any u, $u \texttt{ to } y \texttt{ in } n \mapsto u \texttt{ to } y \texttt{ in } n'$
- Hence, $(u \texttt{ to } y \texttt{ in } n) \star l \mapsto (u \texttt{ to } y \texttt{ in } n') \star l$
- This is $u \star ([y]n \gg l) \mapsto u \star ([y]n' \gg l)$
- $([y]n \gg l) \mapsto_k ([y]n' \gg l)$
- Since $k = ([y]n \gg l)$ we have P.

Closing the Gap

One illustrative case:

In the case $t \texttt{ to } y \texttt{ in } n \mapsto r'$ consider the case $n \mapsto n'$

- For any u, $u \texttt{ to } y \texttt{ in } n \mapsto u \texttt{ to } y \texttt{ in } n'$
- Hence, $(u \texttt{ to } y \texttt{ in } n) \star l \mapsto (u \texttt{ to } y \texttt{ in } n') \star l$
- This is $u \star ([y]n \gg l) \mapsto u \star ([y]n' \gg l)$
- $([y]n \gg l) \mapsto_k ([y]n' \gg l)$
- Since $k = ([y]n \gg l)$ we have P.

Closing the Gap

One illustrative case:

In the case t `to` y `in` $n \mapsto r'$ consider the case $n \mapsto n'$

- For any u, u `to` y `in` $n \mapsto u$ `to` y `in` n'
- Hence, $(u \texttt{ to } y \texttt{ in } n) \star l \mapsto (u \texttt{ to } y \texttt{ in } n') \star l$
- This is $u \star ([y]n \gg l) \mapsto u \star ([y]n' \gg l)$
- $([y]n \gg l) \mapsto_k ([y]n' \gg l)$

$$t \mapsto t' \Longrightarrow t \star k \mapsto t' \star k$$

Closing the Gap

One illustrative case:

In the case $t \texttt{ to } y \texttt{ in } n \mapsto r'$ consider the case $n \mapsto n'$

- For any u, $u \texttt{ to } y \texttt{ in } n \mapsto u \texttt{ to } y \texttt{ in } n'$
- Hence, $(u \texttt{ to } y \texttt{ in } n) \star l \mapsto (u \texttt{ to } y \texttt{ in } n') \star l$
- This is $u \star ([y]n \gg l) \mapsto u \star ([y]n' \gg l)$
- $([y]n \gg l) \mapsto_k ([y]n' \gg l)$
- Since $k = ([y]n \gg l)$ we have P

$$t \star ([y]n \gg L) = (t \texttt{ to } y \texttt{ in } n) \star L$$

Closing the Gap

One illustrative case:

In the case $t \texttt{ to } y \texttt{ in } n \mapsto r'$ consider the case $n \mapsto n'$

- For any u, $u \texttt{ to } y \texttt{ in } n \mapsto u \texttt{ to } y \texttt{ in } n'$
- Hence, $(u \texttt{ to } y \texttt{ in } n) \star l \mapsto (u \texttt{ to } y \texttt{ in } n') \star l$
- This is $u \star ([y]n \gg l) \mapsto u \star ([y]n' \gg l)$
- $([y]n \gg l) \mapsto_k ([y]n' \gg l)$
- Since $k = ([y]n \gg l)$ we have P.

$$K \mapsto_k K' \equiv \forall t.\ t \star K \mapsto t \star K'$$

Closing the Gap

One illustrative case:

In the case $t \texttt{ to } y \texttt{ in } n \mapsto r'$ consider the case $n \mapsto n'$

- For any u, $u \texttt{ to } y \texttt{ in } n \mapsto u \texttt{ to } y \texttt{ in } n'$
- Hence, $(u \texttt{ to } y \texttt{ in } n) \star l \mapsto (u \texttt{ to } y \texttt{ in } n') \star l$
- This is $u \star ([y]n \gg l) \mapsto u \star ([y]n' \gg l)$
- $([y]n \gg l) \mapsto_k ([y]n' \gg l)$
- Since $k = ([y]n \gg l)$ we have P.

$$\bigwedge k'.\ [\![\ k \mapsto k'\ ;\ r = t \star k\ ]\!] \Longrightarrow P$$

CR3 for $T\,\sigma$

- We have everything in place to show:

  (CR3): $\mathit{neutral}(t) \wedge (\forall t'.\, t \mapsto t' \Rightarrow t' \in \mathit{red}_{T\,\tau}) \Rightarrow t \in \mathit{red}_{T\,\tau}$

  Using only (CR1-3) for τ.
- (CR1) and (CR2) are much easier.
- Cases for b and $\sigma \longrightarrow \tau$ could be "imported" from simply-typed λ-calculus.

Looking at it through Isabelle

Isabelle Demo...

Fundamental Theorem

The genaralized case rule can also be applied in the FTLR

lemma to-RED:
assumes m: m $\in$ RED (T σ)
and n : $\forall$ p $\in$ RED σ . n[x::=p] $\in$ RED (T τ)
shows m to x in n $\in$ RED (T τ)

Which boils down to the following **lemma** to-RED-aux:

assumes SN p **and** SN (n[x::=p] $\star$ k) **and** x $\sharp$ p x $\sharp$ k
shows SN (([p] to x in n) $\star$ k)

Fundamental Theorem

The genaralized case rule can also be applied in the FTLR

lemma to-RED:
assumes m: m ∈ RED (T σ)
and n : ∀ p ∈ RED σ . n[x::=p] ∈ RED (T τ)
shows m to x in n ∈ RED (T τ)

Which boils down to the following **lemma** to-RED-aux:

assumes SN p **and** SN (n[x::=p] ⋆ k) **and** x ♯ p x ♯ k
shows SN (([p] to x in n) ⋆ k)

$$t \in red_{T\,\sigma} \equiv \forall K \in red_{\sigma}^{\top}.SN(t \star K)$$
$$K \in red_{\sigma}^{\top} \equiv \forall s \in red_{\sigma}.SN([s] \star K)$$

Fundamental Theorem

lemma to-RED-aux:
assumes SN p **and** SN (n[x::=p] ⋆ k) **and** x ♯ p x ♯ k
shows SN (([p] to x in n) ⋆ k)

Lindley and Stark[2005] show this by (natural) induction on $\max(p) + \max(n \star k) + |k|$.

Fundamental Theorem

lemma to-RED-aux:
assumes SN p **and** SN (n[x::=p] ⋆ k) **and** x ♯ p x ♯ k
shows SN (([p] to x in n) ⋆ k)

Lindley and Stark[2005] show this by (natural) induction on $\max(p) + \max(n \star k) + |k|$.

The $SN(p)$ predicate does not provide a bound $max(p)$

Fundamental Theorem

lemma to-RED-aux:
assumes SN p **and** SN (n[x::=p] ⋆ k) **and** x ♯ p x ♯ k
shows SN (([p] to x in n) ⋆ k)

Lindley and Stark[2005] show this by (natural) induction on $\max(p) + \max(n \star k) + |k|$. I use:

lemma triple-induct:
assumes a: SN (p) **and** b: SN (q)
and hyp: ⋀ (p::trm) (q::trm) (k::stack) .
⟦ ⋀ p′ . p ↦ p′ ⟹ P p′ q k ;
⋀ q′ k . q ↦ q′ ⟹ P p q′ k;
⋀ k′ . |k′| < |k| ⟹ P p q k′ ⟧ ⟹ P p q k
shows P p q k

Fundamental Theorem

lemma to-RED-aux:
assumes SN p **and** SN (n[x::=p] ⋆ k) **and** x ♯ p x ♯ k
shows SN (([p] to x in n) ⋆ k)

Lindley and Stark[2005] show this by (natural) induction on $\max(p) + \max(n \star k) + |k|$. I use:

lemma triple-induct:
assumes a: SN (p) **and** b: SN (q)
and hyp: ⋀ (p::trm) (q::trm) (k::stack) .
⟦ ⋀ p′ q k . ⟦ p ↦ p′ ; SN (q) ⟧⟹ P p′ q k ;
⋀ q′ k . q ↦ q′ ⟹ P p q′ k;
⋀ k′ . |k′| < |k| ⟹ P p q k′ ⟧ ⟹ P p q k
shows P p q k

Fundamental Theorem

lemma to-RED-aux:
assumes SN p **and** SN (n[x::=p] ⋆ k) **and** x ♯ p x ♯ k
shows SN (([p] to x in n) ⋆ k)

Lindley and Stark[2005] show this by (natural) induction on $\max(p) + \max(n \star k) + |k|$. I use:

lemma triple-induct:
assumes a: SN (p) **and** b: SN (q)
and hyp: ⋀ (p::trm) (q::trm) (k::stack) .
⟦ ⋀ p′ . p ↦ p′ ⟹ P p′ q k ;
⋀ q′ k . q ↦ q′ ⟹ P p q′ k;
⋀ k′ . |k′| < |k| ⟹ P p q k′ ⟧ ⟹ P p q k
shows P p q k

Digression – Reverse Stack Induction Rule

The standard rule:

$$\frac{\bigwedge y\ n\ L\ z.\ [\![y \sharp z;\ y \sharp L;\ \bigwedge z.\ P\ z\ L]\!] \Longrightarrow P\ z\ (St\ y\ n\ L) \qquad \bigwedge z.\ P\ z\ Id}{P\ z\ K}$$

The reverse rule:

$$\frac{\bigwedge y\ n\ L\ z.\ [\![y \sharp z;\ y \sharp L;\ \bigwedge z.\ P\ z\ L]\!] \Longrightarrow P\ z\ (L +\!\!+ St\ y\ n\ Id) \qquad \bigwedge z.\ P\ z\ Id}{P\ z\ K}$$

Application

Lemma

$$K \mapsto_k K' \Rightarrow \mathit{length}\ K \geq \mathit{length}\ K'$$

where $K \mapsto_k K' \equiv \forall t.\ t \star K \mapsto t \star K'$

Proof on Paper: "Suppose $x \star K \mapsto x \star K'$
and $K = (y_1)n_1 :: (y_2)n_2 :: \ldots :: \mathit{Id}$"
"There are only two reductions that might change the length of K"

Isabelle/HOL-Nominal: Roughly 90 lines inductive proof

Application

Lemma

$$K \longmapsto_1 K' \Rightarrow \mathit{length}\ K > \mathit{length}\ K'$$

We don't need this!

and $K = (y_1)n_1 :: (y_2)n_2 :: \ldots :: Id$"

"There are only two reductions that might change the length of K"

Isabelle/HOL-Nominal: Roughly 90 lines inductive proof

Reverse Stack Induction Rule

The standard rule:

$$\bigwedge z.\ P\ z\ Id$$

$$\bigwedge y\ n\ L\ z.\ [\![y \sharp z;\ y \sharp L;\ \bigwedge z.\ P\ z\ L]\!] \Longrightarrow P\ z\ (L +\!\!+ St\ y\ n\ Id)$$

$$\overline{P\ z\ K}$$

We don't need this either!

$$\bigwedge z.\ P\ z\ Id$$

Nominal?

How much did I profit from the nominal package?

- Basic definitions straightforward:
  - Terms, Types, Substitution, Reductions, Typing,
  - Require more freshness conditions than one uses on paper
  - Variable convention also in rule inductions and strong inversion
- Very low coding gap
  - Lemmas can be stated exactly as on paper
- Additional Freshness conditions sometimes cumbersome and require alpha renaming.

Faithful?

How faithful is the formalization?

- Reduction stated with additional freshness conditions
  - I have shown adequacy
- Different characterization of strong normalization
  - I have also shown adequacy
- The lemmas/cases spelled out in Lindley Stark[2005]:
  - rather close / one has to deal with freshness conditions
- Need many additional lemmas
  - Most have two line proofs using `nominal_induct` and `auto`
  - Notable Exception: case rule on $t \star k \mapsto r$

References


T.Nipkow, L.C. Paulson, M. Wenzel A Proof Assistant for Higher-Order Logic, http://isabelle.in.tum.de/dist/Isabelle/doc/tutorial.pdf


T.Nipkow, A Tutorial Introduction to Structured Isar Proofs http://isabelle.in.tum.de/dist/Isabelle/doc/isar-overview.pdf


C. Urban, Nominal Techniques in Isabelle/HOL, Journal of Automatic Reasoning, Vol. 40(4), pp: 327-356, 2008


S. Lindley and I. Stark, Reducibility and TT-lifting for Computation Types, Typed Lambda Calculi and Applications, LNCS Vol. 3461, pp: 262-277, Springer-Verlag, 2005.

Thank You!