Proving PSN after ruining a perfectly good calculus

Shane Ó Conchúir

December 6, 2006

Abstract

We prove that a modified version of Kesner and Lengrand's λlxr calculus has the property of preservation of strong normalisation (PSN) of β-reduction. The proof uses a general technique due to Lengrand and a slight adaptation of his proof of PSN for λlxr. In other work, our proof will be used to prove PSN for another calculus.

Introduction

λlxr [6, 5] is an explicit substitution calculus which is a sound and complete computational counterpart to the intuitionistic part of the Proof Nets of Linear Logic [4]. It is also the first published explicit substitution calculus to our knowledge to enjoy the properties of confluence, preservation of strong normalisation (PSN), and full composition of substitutions.

λlxr builds partly on work by David and Guillaume on the λ_{ws} calculus [2]. The λ_{ws} calculus allowed a level of composition of substitutions whilst retaining PSN and was one of the first explicit substitution calculi which satisfied step-by-step simulation of β-reduction, confluence on terms with metavariables, and PSN.

Terms in λlxr are linear and weakenings and contractions are used to allow this. This linearity avoids many problems where composition of substitutions usually break PSN such as the needless copying of an explicit substitution. Fernández and Mackie [3] explored these notions in earlier work.

In this paper, we modify one of the reduction rules of λlxr to define a new calculus λblxr and prove that it also has the PSN property. This latter calculus is not interesting in its own right but we show in other work [12] that it can simulate reduction of another calculus Λ_{sub} due to Milner [10]. We then use λblxr to prove PSN for this calculus.

In Section 1, we summarise the λlxr calculus and introduce λblxr. The proof of PSN is presented in Section 2. The proof is based on Lengrand's general strategy for proving PSN through simulation in λI (extended with a 'memory construct') and his proof of PSN for λlxr [9]. Our modification of λlxr could be described as *ruining* the calculus as we make it unnecessarily more complicated. Interestingly, we also have to introduce some inelegance into the original proofs to prove PSN for λblxr.

1 Summary of λlxr

We only summarise the details of λlxr necessary for the proof. The reader is referred to the original works [6, 5] for a proper introduction.

The set of terms of λlxr is defined (with a slight change of notation) by

$$t ::= x \mid \lambda x.t \mid tt \mid t\langle x := t\rangle \mid W_x(t) \mid C_x^{y,z}(t)$$

The constructor $t\langle x := u\rangle$ denotes an explicit substitution à la λxgc. The constructor $W_x(t)$ is an *explicit weakening* and the constructor $C_x^{y,z}(t)$ is an *explicit contraction*. The sets of free variables for the first four constructors are as expected. x is free in $W_x(t)$ and $C_x^{y,z}(t)$ whereas y and z are bound in the latter. We follow a variable convention where each bound name of a term t is distinct and different from any free names in t. We denote the free variables of a term t by $\mathrm{FV}(t)$ and the bound variables by $\mathrm{BV}(t)$.

We now discuss three important features in λlxr – weakenings, contractions, and linearity of terms.

The term $W_x(t)$ is an annotated form of t which states that the free variable x does not occur free in t. As it is explicitly part of the syntax, it can play a rôle in the reduction relation of λlxr and weakenings are in fact used to provide an explicit garbage collection rule. Consider the term $W_x(t)\langle x := u\rangle$. As x does not occur free in t, we may want to garbage collect the substitution. The rule ($Weak1$) in Figure 2 does precisely this. Weakenings in λlxr may always be pulled out to the top level, allowing efficient garbage collection.

Substitution in λlxr is defined with a set of distributive rules. Weakenings also allow efficient propagation of substitutions. For example, propagating the substitution $x := u$ through $W_x(t)$ is pointless as no substitution can take place and so the reduction rules do not permit this propagation.

Weakenings allow free variables to be kept through reduction. The two destructive rules are (Var) and ($Weak1$). As expected, the substitution rule (Var) does not lose free variables. Interestingly, the garbage collection rule ($Weak1$) remembers the free variables of the discarded substitution via a weakening. Kesner and Lengrand compare this preservation of free variables to "interface preserving" [8] in interaction nets.

Contractions in λlxr allow the linearity of terms discussed below. The term $C_x^{y,z}(t)$ may be read as 't where y and z are x.'

Terms in λlxr may always be assumed to be linear. A term t is linear if "*in every subterm, every variable has at most one free occurrence, and every binder binds a variable that does have a free occurrence (and hence only one)*" [6]. It is possible to translate every λ-term to a (linear) λlxr term. This linearity appears to be a large factor in allowing λlxr to retain PSN whilst having full composition of substitutions (FCS). Substitutions are also never needlessly copied – the ($Cont1$) rule which copies substitutions in λlxr does so conditionally and out of need.

The congruence axioms and reduction rules for λlxr can be found in Figures 1 and 2 respectively. Rewriting in λlxr is performed using the reduction rules modulo the smallest congruence generated by the axioms. The congruence axioms were chosen to strengthen the relationship between λlxr and Proof Nets. In the reduction rules, the notation $R_\Delta^\Phi(t)$, where Φ and Δ are finite lists (with no repetition) of distinct variables and equal length, denotes the result of

$$
\begin{array}{llll}
C_w^{x,v}(C_x^{z,y}(t)) & \equiv_A & C_w^{x,y}(C_x^{z,v}(t)) & \text{if } x \neq y, v \\
C_x^{y,z}(t) & \equiv_{C1_c} & C_x^{z,y}(t) & \\
C_{x'}^{y',z'}(C_x^{y,z}(t)) & \equiv_{C2_c} & C_x^{y,z}(C_{x'}^{y',z'}(t)) & \text{if } x \neq y', z' \& x' \neq y, z \\
W_x(W_y(t)) & \equiv_{C_w} & W_y(W_x(t)) & \\
t\langle x := v\rangle\langle y := u\rangle & \equiv_S & t\langle y := u\rangle\langle x := v\rangle & \text{if } y \notin \mathrm{FV}(v) \& x \notin \mathrm{FV}(u) \\
 & & & \& \; x \neq y \\
C_w^{y,z}(t)\langle x := v\rangle & \equiv_{Cont2} & C_w^{y,z}(t\langle x := v\rangle) & \text{if } x \neq w \& y, z \notin \mathrm{FV}(v)
\end{array}
$$

Figure 1: Congruences for λlxr

$$
\begin{array}{llll}
(\lambda x.t)u & \longrightarrow_B & t\langle x := u\rangle &
\end{array}
$$

System x

$$
\begin{array}{llll}
(\lambda y.t)\langle x := u\rangle & \longrightarrow_{Abs} & \lambda y.t\langle x := u\rangle & \\
(tu)\langle x := P\rangle & \longrightarrow_{App1} & t\langle x := P\rangle u & x \in \mathrm{FV}(t) \\
(tu)\langle x := P\rangle & \longrightarrow_{App2} & tu\langle x := P\rangle & x \in \mathrm{FV}(u) \\
x\langle x := t\rangle & \longrightarrow_{Var} & t & \\
W_x(t)\langle x := u\rangle & \longrightarrow_{Weak1} & W_{\mathrm{FV}(u)}(t) & \\
W_y(t)\langle x := u\rangle & \longrightarrow_{Weak2} & W_y(t\langle x := u\rangle) & x \neq y \\
t\langle x := P\rangle\langle y := Q\rangle & \longrightarrow_{Comp} & t\langle x := P\langle y := Q\rangle\rangle & y \in \mathrm{FV}(P) \\
 & & & \\
C_x^{y,z}(t)\langle x := u\rangle & \longrightarrow_{Cont1} & C_\Phi^{\Delta,\Pi}(t\langle y := u_1\rangle\langle z := u_2\rangle) & \text{where} \\
 & & & \Phi := \mathrm{FV}(u) \\
 & & & u_1 = R_\Delta^\Phi(u) \\
 & & & u_2 = R_\Pi^\Phi(u)
\end{array}
$$

System r

$$
\begin{array}{llll}
\lambda x.W_y(t) & \longrightarrow_{W\ Abs} & W_y(\lambda x.t) & x \neq y \\
W_y(t)u & \longrightarrow_{W\ App1} & W_y(tu) & \\
tW_y(u) & \longrightarrow_{W\ App2} & W_y(tu) & \\
t\langle x := W_y(u)\rangle & \longrightarrow_{W\ Subs} & W_y(t\langle x := u\rangle) & \\
C_w^{y,z}(W_y(t)) & \longrightarrow_{terge} & R_w^z(t) & \\
C_w^{y,z}(W_x(t)) & \longrightarrow_{Cross} & W_x(C_w^{y,z}(t)) & x \neq y, z \\
C_w^{y,z}(\lambda x.t) & \longrightarrow_{C\ Abs} & \lambda x.C_w^{y,z}(t) & \\
C_w^{y,z}(tu) & \longrightarrow_{C\ App1} & C_w^{y,z}(t)u & y, z \in \mathrm{FV}(t) \\
C_w^{y,z}(tu) & \longrightarrow_{C\ App2} & tC_w^{y,z}(u) & y, z \in \mathrm{FV}(u) \\
C_w^{y,z}(t\langle x := u\rangle) & \longrightarrow_{C\ Subs} & t\langle x := C_w^{y,z}(u)\rangle & y, z \in \mathrm{FV}(u)
\end{array}
$$

Figure 2: Reduction rules for λlxr

simultaneously replacing $x \in \Phi$ in t with $y \in \Delta$ where both variables occur as the i^{th} variable in their respective lists. The meta-notation $W_{\mathrm{FV}(u)}$ and $C_{\Phi}^{\Delta,\Pi}$ denotes multiple weakenings and contractions – the order is irrelevant up to congruences $\equiv_{C2_c}$ and $\equiv_{C_w}$.

Many of the reduction rules of λlxr (especially in System r) deal with pulling weakenings outwards and pushing contractions inwards. Linearity of terms means that substitutions are not replicated during propagation through a term unless a contraction is reached in which case the substitution is duplicated and the copies renamed to maintain linearity ($Cont1$). Besides these rules, the main familiar ones are substitution introduction (B), copying (Var), and explicit garbage collection ($Weak1$). There is one reduction rule for explicit composition of substitutions ($Comp$). This rule only takes care of the case $y \in \mathrm{FV}(P)$ but the other case $y \in \mathrm{FV}(t)$ is taken care of by the $\equiv_S$ congruence (assuming linearity and the variable convention). This allows λlxr FCS.

Lemma 1. $\longrightarrow_{\mathrm{xr}}$ *is strongly normalising [5].*

1.1 The modified calculus

Our sole modification to λlxr is by replacing the rule used to create explicit substitutions from β-redexes, $\longrightarrow_B$. We replace it with a rule which creates two explicit substitutions instead of one where the outer substitution is always garbage *i.e.* there is no free occurrence of the variable that the substitution binds in the term.

Definition 2 ($\longrightarrow_{Bs}$). *The reduction $\longrightarrow_{Bs}$ is defined as the contextual closure, modulo $\equiv$, of the rule*

$$(\lambda x.t)u \longrightarrow_{Bs} C_{\Theta}^{\Gamma,\Psi}\big(\big(W_{x'}(t\langle x := R_{\Gamma}^{\Theta}(u)\rangle)\big)\langle x' := R_{\Psi}^{\Theta}(u)\rangle\big)$$

where $\Theta = \mathrm{FV}(u)$ and x' is a fresh name.

The outer substitution $\langle x' := R_{\Psi}^{\Theta}(u)\rangle$ in the rule is always garbage. This seems odd – why create a garbage copy of a substitution? We require this property to prove PSN for a calculus Λ_{sub} due to Milner [10]. The calculus has the notion of 'non-local substitution' where an explicit substitution is not propagated through a term but remains in place while being linked to the free occurrences of its bound variable below. This property is due to the bigraphical design of Λ_{sub}. Our proof of PSN for Λ_{sub} is based on simulating Λ_{sub} reductions in a λlxr-like calculus. In other work [12], we explain that in order for a simulation to work, we require two copies of a substitution in the translation from Λ_{sub} to λlxr – one is garbage and immobile, which provides a sort of syntactic match between a term and its translation, and the other is allowed to propagate through the term to simulate substitution. To simulate the creation of explicit substitutions, we require the $\longrightarrow_B$ rule to be replaced as above.

Definition (λblxr). *We let* λblxr *denote the calculus obtained from* λlxr *by replacing $\longrightarrow_B$ with $\longrightarrow_{Bs}$. We let $\longrightarrow_{\lambda\mathrm{blxr}}$ denote the reduction relation of* λblxr. *$\longrightarrow^*_{\lambda\mathrm{blxr}}$ denotes the reflexive and transitive closure of $\longrightarrow_{\lambda\mathrm{blxr}}$.*

Again, rewriting is modulo the congruence $\equiv$. λblxr is clearly less elegant than λlxr – we have ruined it by adding in unnecessary substitutions. However, we are interested in it as a means to a proof, not in itself. For us, it is only important that it has the PSN property.

2 Proving PSN

We now begin the proof of PSN for λblxr. It seems reasonable that the property holds as λblxr differs from λlxr only in the $\longrightarrow_{Bs}$ rule which creates two substitutions; one as normal and one which is 'garbage' and can not propagate through the term. Intuitively, this does not seem to introduce new cases of infinite reductions as the garbage substitution may only interact with substitutions above it as the normal substitution can.

Our proof uses Lengrand's strategy of proving PSN by simulating reduction of a calculus in a version of λI with memory [7][1]. The proof is unsurprisingly similar to the proof of PSN for λlxr [9]. It transpires that just as we have ruined λlxr, we will also have to introduce some inelegance into some of the relations and encodings that Lengrand defines.

For notational convenience, we denote the λI calculus with memory simply as λI and refer to the original λI calculus [1] as Church's λI calculus.

Notation. *When discussing a reduction system with reduction relation R,* SN_R *denotes the set of strongly normalising terms and* WN_R *denotes the set of weakly normalising terms.* $\longrightarrow_R^*$ *and* $\longrightarrow_R^+$ *denote the reflexive and transitive closure of R respectively.*

Definition 3. *The set Λ_I of terms of the λI-calculus is defined by*

$$T, U ::= x \mid \lambda x.T \mid (TU) \mid [T, U]$$

where $x \in \mathrm{FV}(T)$ for all abstractions $\lambda x.T$.

Notation (λI-terms). $[U, T_1, T_2, \ldots, T_n]$ *or* $[U, \overrightarrow{T_i}]$ *denote the term* $[\ldots[[U, T_1], T_2], \ldots, T_n]$. $\overrightarrow{T_i}$ *denotes the application* $T_1 \ldots T_n$.

Notation (λI-contexts). $C[\,]$ *denotes a context with a hole. In this work,* $C[M]$ *denotes the result of filling the hole of a context with a term M* such that no free variables of M are bound by the context.

As usual, $T\{x \setminus U\}$ denotes the term T where all free occurences of x are replaced by U. Again, we follow a variable convention so that variable capture is avoided.

Lemma 4 (Substitutions). *Given any terms $T, U, V \in \Lambda_I$, we have $T\{x \setminus U\} \in \Lambda_I$ and $T\{x \setminus U\}\{y \setminus V\} = T\{y \setminus V\}\{x \setminus U\{y \setminus V\}\}$ so long as there is no variable capture.*

The reduction rules of λI are:

$$\begin{array}{lrcl} (\beta) & (\lambda x.T)U & \longrightarrow & T\{x \setminus U\} \\ (\pi) & [T, V]U & \longrightarrow & [TU, V] \end{array}$$

Proposition 5 (Church's Theorem). *In λI,*

$$T \in \mathrm{SN}_{\beta\pi} \Leftrightarrow T \in \mathrm{WN}_{\beta\pi} \Leftrightarrow \forall S \subseteq T, S \in \mathrm{WN}_{\beta\pi}.$$

[1]This is referred to as Church-Klop's λI-calculus in [9] and denoted as $\lambda I_{[\,,\,]}$ in Klop's thesis [7].

Proof. λI is a substructure of a definable extension of Church's λI calculus. The proof follows from Klop's thesis [7, Corollary I.7.5].

(More generally, λI is a regular/orthogonal, non-erasing combinatory reduction system [7]. Klop provides a further generalisation of Church's Theorem for this case [Ibid., Theorem 5.9.3].) $\square$

2.1 Proof strategy

The notion of PSN is defined for any calculus which extends the syntax of the λ-calculus. λlxr (and hence λblxr) is not an extension as the terms are required to be linear. PSN is defined for λlxr as meaning that "every strongly normalisable λ-term is *encoded* into a strongly normalisable λlxr-term" [5]. We adopt this definition and Lengrand's proof strategy for our proof. The proof strategy is as follows. In Section 2.2:

1. We define a relation $\mathcal{J}$ between λblxr-terms and λI-terms.
2. We prove that $\longrightarrow_{\mathrm{xr}}$ is weakly simulated through $\mathcal{J}$ and $\longrightarrow_{Bs}$ is strongly simulated through $\mathcal{J}$. Thus, strong normalisation is reflected back through the relation w.r.t. $\longrightarrow_{\lambda\mathrm{blxr}}$ and $\longrightarrow_{\beta\pi}$.

In Section 2.3:

3. We define an encoding $\mathcal{A}$ from λ-terms to λblxr-terms.
4. We define an encoding j from λ-terms to λI-terms.
5. We prove that $\mathcal{A}(t)\ \mathcal{J}\ j(t)$.

In Section 2.4:

6. We prove that j preserves strong normalisation w.r.t. $\longrightarrow_{\beta}$ and $\longrightarrow_{\beta\pi}$.

We then conclude PSN: given any strongly normalising λ-term t, $j(t)$ is strongly normalising (step 6) and the λblxr encoding $\mathcal{A}(t)$ is related to $j(t)$ (step 5). As strong normalisation is reflected back through $\mathcal{J}$ (step 2), it follows that $\mathcal{A}(t)$ is strongly normalising.

The general proof strategy is depicted quite succinctly by Lengrand [9].

2.2 Simulation of λblxr in λI

The proof of PSN for λlxr used a relation between λlxr terms and Λ_I.

Definition 6. *The relation $\mathcal{I}$ between well-formed λlxr-terms and Λ_I is given by the following rules:*

$$\frac{}{x\ \mathcal{I}\ x} \qquad \frac{t\ \mathcal{I}\ T}{\lambda x.t\ \mathcal{I}\ \lambda x.T} \qquad \frac{t\ \mathcal{I}\ T \qquad u\ \mathcal{I}\ U}{tu\ \mathcal{I}\ TU} \qquad \frac{t\ \mathcal{I}\ T}{t\ \mathcal{I}\ [T, N]}\ N \in \Lambda_I$$

$$\frac{t\ \mathcal{I}\ T \qquad u\ \mathcal{I}\ U}{t\langle x := u\rangle\ \mathcal{I}\ T\{x \setminus U\}} \qquad \frac{t\ \mathcal{I}\ T}{C_x^{y,z}(t)\ \mathcal{I}\ T\{y \setminus x\}\{z \setminus x\}} \qquad \frac{t\ \mathcal{I}\ T}{W_x(t)\ \mathcal{I}\ T}\ x \in \mathrm{FV}(T)$$

An interesting property of this relation is that it is non-deterministic. The fourth rule above allows arbitrary Λ_I terms to be 'tacked onto' a Λ_I-term T which is related to some λlxr-term t. This non-determinism has to be accounted for in the various proofs which is where the vector notation $\overrightarrow{N}$ comes in handy.

Example 7 (non-determinism and proof trees). *Let $t \mathcal{I} T$ and $u \mathcal{I} U$. The Λ_I-terms which are related to $(\lambda x.t)\langle y := u\rangle$ take the form $[(\lambda x.T)\{y \setminus U\}, \overrightarrow{M}]$ as seen by the proof tree below.*

$$\dfrac{\dfrac{\dfrac{t \mathcal{I} T}{\lambda x.t \mathcal{I} \lambda x.T}}{\lambda x.t \mathcal{I} [\lambda x.T, \overrightarrow{M}']} \qquad u \mathcal{I} U}{(\lambda x.t)\langle y := U\rangle \mathcal{I} [\lambda x.T, \overrightarrow{M}']\{y \setminus U\}}$$

This last λI-term is equivalent to $[(\lambda x.T)\{y \setminus U\}, \overrightarrow{M}]$.

Generally, we will not care what the contents of $\overrightarrow{M}$ actually are. They represent some arbitary terms added in by the relation.

This non-determinism is frequently employed to derive the terms related to λlxr terms containing weakenings. Consider the term $W_x(t)$ where $x \notin t$ by linearity. Let $t \mathcal{I} T$ and $x \notin \mathrm{FV}(T)$. In order to find a term related to $W_x(t)$ by $\mathcal{I}$, we first have to introduce a free x into T. This may be done as follows.

$$\dfrac{\dfrac{t \mathcal{I} T}{t \mathcal{I} [T, x]}}{W_x(t) \mathcal{I} [T, x]}$$

This relation $\mathcal{I}$ turns out not to be sufficient for our proof that $\longrightarrow_{Bs}$ is strongly simulated through the relation normalisation.

Proposition 8. *$\longrightarrow_{Bs}$ is not strongly simulated by $\longrightarrow^+_{\beta\pi}$ through $\mathcal{I}$.*

Proof. We will give a counterexample. Let

$$t \equiv (\lambda x.p)u, \quad p \mathcal{I} P, \quad u \mathcal{I} U, \quad \text{and } t \mathcal{I} [[\lambda x.P, \overrightarrow{R}]U, \overrightarrow{S}] \equiv T.$$

Let $x' \notin \mathrm{FV}(t) \cup \mathrm{BV}(t) \cup \mathrm{FV}(\overrightarrow{R})$. Now consider

$$t \equiv (\lambda x.p)u \longrightarrow_{Bs} C^{\Gamma,\Psi}_{\Theta}\big(\big(W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\big)\langle x' := R^{\Theta}_{\Psi}(u)\rangle\big) \equiv t'$$

where $\Theta = \mathrm{FV}(u)$. We will show that there is $T' \in \Lambda_I$ such that $t' \mathcal{I} T'$ and T cannot $\longrightarrow^+_{\beta\pi}$-reduce to T'.

By [9], $R^{\Theta}_{\Gamma}(u) \mathcal{I} R^{\Theta}_{\Gamma}(U)$ and $R^{\Theta}_{\Psi}(u) \mathcal{I} R^{\Theta}_{\Psi}(U)$. The general form of T' is given by the proof tree in Figure 3 where the subterms $\overrightarrow{M}^i$ and $\overrightarrow{N}^j$ are added by the fourth rule of Definition 6 (a change in superscript denotes a different subterm occuring from further applications of this rule or by substitution). In particular, T' contains an occurrence of U outside of $P\{x \setminus U\}$, $\overrightarrow{M}$, or $\overrightarrow{N}$.

We will try to simulate the reduction $t \longrightarrow_{Bs} t'$ in λI by starting with $T \equiv [[\lambda x.P, \overrightarrow{R}]U, \overrightarrow{S}] \longrightarrow^+_{\pi} [(\lambda x.P)U, \overrightarrow{R}, \overrightarrow{S}] \longrightarrow_{\beta} [P\{x \setminus U\}, \overrightarrow{R}, \overrightarrow{S}]$. It is not always true that U is not a subterm of $\overrightarrow{R}$ or $\overrightarrow{S}$. However, in this case, the $\longrightarrow_{Bs}$ reduction cannot be simulated as U is always present outside of $P\{x \setminus U\}$ in T'. □

This counterexample is sufficient to disallow the relation $\mathcal{I}$ for our purposes. The problem is that our $\longrightarrow_{Bs}$ rule creates two copies of an explicit substitution whenever it fires. The relation $\mathcal{I}$ then always introduces U as a subterm of the Λ_I term corresponding to the garbage substitution bounded by the fresh x'. Our solution is simply to add some redundancy into the relation $\mathcal{I}$.

$$
\begin{array}{c}
t\ \mathcal{I}\ T \qquad R^{\Theta}_{\Gamma}(u)\ \mathcal{I}\ R^{\Theta}_{\Gamma}(U) \\ \hline
p\langle x := R^{\Theta}_{\Gamma}(u)\rangle\ \mathcal{I}\ P\{x \setminus R^{\Theta}_{\Gamma}(U)\} \\ \hline
p\langle x := R^{\Theta}_{\Gamma}(u)\rangle\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^1] \\ \hline
p\langle x := R^{\Theta}_{\Gamma}(u)\rangle\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^1, C[x']] \\ \hline
p\langle x := R^{\Theta}_{\Gamma}(u)\rangle\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^1, C[x'], \overrightarrow{N}^1] \\ \hline
W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^1, C[x'], \overrightarrow{N}^1] \\ \hline
W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^1, C[x'], \overrightarrow{N}^2] \\ \hline
W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\langle x' := R^{\Theta}_{\Psi}(u)\rangle\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^2, C[R^{\Theta}_{\Psi}(U)], \overrightarrow{N}^3] \\ \hline
W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\langle x' := R^{\Theta}_{\Psi}(u)\rangle\ \mathcal{I}\ [P\{x \setminus R^{\Theta}_{\Gamma}(U)\}, \overrightarrow{M}^2, C[R^{\Theta}_{\Psi}(U)], \overrightarrow{N}^4] \\ \hline
C^{\Gamma,\Psi}_{\Theta}(W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\langle x' := R^{\Theta}_{\Psi}(u)\rangle)\ \mathcal{I}\ [P\{x \setminus U\}, \overrightarrow{M}, C[U], \overrightarrow{N}^5] \\ \hline
C^{\Gamma,\Psi}_{\Theta}(W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\langle x' := R^{\Theta}_{\Psi}(u)\rangle)\ \mathcal{I}\ [P\{x \setminus U\}, \overrightarrow{M}, C[U], \overrightarrow{N}]
\end{array}
$$

Figure 3: The general term related to t'

Definition 9. *The relation $\mathcal{J}$ between well-formed* λblxr*-terms and Λ_I is given by the following rules:*

$$
\frac{}{x\ \mathcal{J}\ x}
\qquad
\frac{t\ \mathcal{J}\ T}{\lambda x.t\ \mathcal{J}\ \lambda x.[T, x]}
\qquad
\frac{t\ \mathcal{J}\ T \qquad u\ \mathcal{J}\ U}{tu\ \mathcal{J}\ TU}
\qquad
\frac{t\ \mathcal{J}\ T}{t\ \mathcal{J}\ [T, N]}\ N \in \Lambda_I
$$

$$
\frac{t\ \mathcal{J}\ T \qquad u\ \mathcal{J}\ U}{t\langle x := u\rangle\ \mathcal{J}\ T\{x \setminus U\}}
\qquad
\frac{t\ \mathcal{J}\ T}{C^{y,z}_{x}(t)\ \mathcal{J}\ T\{y \setminus x\}\{z \setminus x\}}
\qquad
\frac{t\ \mathcal{J}\ T}{W_x(t)\ \mathcal{J}\ T}\ x \in \mathrm{FV}(T)
$$

The only rule we have changed is the one for abstractions. We require that an abstraction in a related Λ_I term carries around a free occurrence of x 'tacked on.' This is a redundant feature since $x \in \mathrm{FV}(T)$ in the rule (Lengrand shows that $x \in \mathrm{FV}(t)$ and $t\ \mathcal{I}\ T$ implies $x \in \mathrm{FV}(T)$ [9]) but we require this redundancy, having introduced it into λblxr.

We can now prove the following properties of $\mathcal{J}$. All proofs here (and subsequent proofs in the following sections) are adapted from the original work [9]. For brevity, we omit some cases when they are already dealt with in that work. When we say that a proof 'remains the same' we mean that the original proof (for the original encodings/relations) suffices for any alterations we have made.

Lemma 10. *If $t\ \mathcal{J}\ M$, then*

1. $\mathrm{FV}(t) \subseteq \mathrm{FV}(M)$
2. $M \in \Lambda_I$
3. $x \notin \mathrm{FV}(t)$ *and* $N \in \Lambda_I$ *implies* $t\ \mathcal{J}\ M\{x \setminus N\}$
4. $t \equiv t'$ *implies* $t'\ \mathcal{J}\ M$
5. $R^{\Gamma}_{\Delta}(t)\ \mathcal{J}\ R^{\Gamma}_{\Delta}(M)$

Proof. Property (1) is proved by induction on the proof tree. The abstraction rule does not add any free variables as $x \in \mathrm{FV}(t)$ and so $x \in \mathrm{FV}(T)$ by the inductive hypothesis. For this case, we have

$$\mathrm{FV}(\lambda x.t) = \mathrm{FV}(t) \setminus \{x\} \subseteq^{I.H.} \mathrm{FV}(T) \setminus \{x\} = \mathrm{FV}(\lambda x.[T, x]).$$

Property (2) is also proved by induction, using Lemma 4. The new case is shown by $T \in \Lambda_I \Rightarrow [T, x] \in \Lambda_I \Rightarrow \lambda x.[T, x] \in \Lambda_I$. Properties (3) and (5) are proved by induction, noting that we do not add any new free variables with our rule. The proof of Property (4) remains the same as abstractions are not involved in any of the congruences. ☐

Theorem 11 (Simulation in λI).

1. *If $t \ \mathcal{J}\ T$ and $t \longrightarrow_{\mathrm{xr}} t'$, then $t' \ \mathcal{J}\ T$.*
2. *If $t \ \mathcal{J}\ T$ and $t \longrightarrow_{Bs} t'$, then there is $T' \in \Lambda_I$ such that $t' \ \mathcal{J}\ T'$ and $T \longrightarrow^{+}_{\beta\pi} T'$.*

Proof. We only consider the cases at the root affected by our change to $\mathcal{I}$. The closure under context follows as in the original proof. In the following, $p \ \mathcal{J}\ P$ and $u \ \mathcal{J}\ U$.

– Let $t \equiv (\lambda x.p)u \longrightarrow_{Bs} C^{\Gamma,\Psi}_{\Theta}\big(\big(W_{x'}(p\langle x := R^{\Theta}_{\Gamma}(u)\rangle)\big)\langle x' := R^{\Theta}_{\Psi}(u)\rangle\big) \equiv t'$ where $\Theta = \mathrm{FV}(p)$.

$$\begin{array}{rcl} T & = & [[\lambda x.[P, x], \overrightarrow{R}]U, \overrightarrow{S}] \\ & \longrightarrow^{+}_{\pi} & [\lambda x.[P, x]U, \overrightarrow{R}, \overrightarrow{S}] \\ & \longrightarrow_{\beta} & [P\{x \setminus U\}, U, \overrightarrow{R}, \overrightarrow{S}] \end{array}$$

Figure 3 shows that t' is related to this last term, replacing $\overrightarrow{M}$ with the empty term, C with the empty context, and $\overrightarrow{N}$ with $\overrightarrow{R}$ and $\overrightarrow{S}$.

– Let $t \equiv (\lambda y.p)\langle x := u\rangle \longrightarrow_{Abs} \lambda y.p\langle x := u\rangle \equiv t'$. By convention, $y \neq x$. $T = [[\lambda y.[P, y], \overrightarrow{M}]\{x \setminus U\}, \overrightarrow{N}] = [\lambda y.[P\{x \setminus U\}, y], \overrightarrow{M}\{x \setminus U\}, \overrightarrow{N}]$. This term is related to t' as shown below.

$$\dfrac{\dfrac{\dfrac{p \ \mathcal{J}\ P \qquad u \ \mathcal{J}\ U}{p\langle x := u\rangle \ \mathcal{J}\ P\{x \setminus U\}}}{\lambda y.p\langle x := u\rangle \ \mathcal{J}\ \lambda y.[P\{x \setminus U\}, y]}}{\lambda y.p\langle x := u\rangle \ \mathcal{J}\ [\lambda y.[P\{x \setminus U\}, y], \overrightarrow{M}\{x \setminus U\}, \overrightarrow{N}]}$$

– Let $t \equiv \lambda x.W_y(p) \longrightarrow_{WAbs} W_y(\lambda x.p) \equiv t'$. By linearity, $y \notin \mathrm{FV}(p)$. $T = [\lambda x.[P, C[y], \overrightarrow{M}, x], \overrightarrow{N}]$. This term is related to t' as shown below.

$$\dfrac{\dfrac{\dfrac{\dfrac{p \ \mathcal{J}\ P}{p \ \mathcal{J}\ [P, C[y], \overrightarrow{M}]}}{\lambda x.p \ \mathcal{J}\ \lambda x.[P, C[y], \overrightarrow{M}, x]}}{W_y(\lambda x.p) \ \mathcal{J}\ \lambda x.[P, C[y], \overrightarrow{M}, x]}}{W_y(\lambda x.p) \ \mathcal{J}\ [\lambda x.[P, C[y], \overrightarrow{M}, x], \overrightarrow{N}]}$$

– Let $t \equiv C_w^{y,z}(\lambda x.p) \longrightarrow_{CAbs} \lambda x.C_w^{y,z}(p) \equiv t'$. By linearity, $y, z \neq x$.

$$\begin{array}{rcl} T & = & [[\lambda x.[P,x],\overrightarrow{M}]\{y \setminus w\}\{z \setminus w\},\overrightarrow{N}] \\ & = & [\lambda x.[P\{y \setminus w\}\{z \setminus w\},x],\overrightarrow{M}\{y \setminus w\}\{z \setminus w\},\overrightarrow{N}] \end{array}$$

This term is related to t' as shown below.

$$\dfrac{\dfrac{\dfrac{p \mathrel{\mathcal{J}} P}{C_w^{y,z}(p) \mathrel{\mathcal{J}} P\{y \setminus w\}\{z \setminus w\}}}{\lambda x.C_w^{y,z}(p) \mathrel{\mathcal{J}} \lambda x.[P\{y \setminus w\}\{z \setminus w\},x]}}{\lambda x.C_w^{y,z}(p) \mathrel{\mathcal{J}} [\lambda x.[P\{y \setminus w\}\{z \setminus w\},x],\overrightarrow{M}\{y \setminus w\}\{z \setminus w\},\overrightarrow{N}]}$$

□

Corollary 12. *If $t \mathrel{\mathcal{J}} T$ and $T \in \mathrm{SN}_{\beta\pi}$ then $t \in \mathrm{SN}_{\lambda\mathrm{blxr}}$.*

Proof. Proof by contradiction [9] based on the termination of $\longrightarrow_{\mathrm{xr}}$ (Lemma 1) and Theorem 11. □

2.3 Encoding the λ-calculus in λI and λlxr

Kesner and Lengrand give an encoding of the λ-calculus in λlxr.

Definition 13 ([5]). *The encoding of λ-terms is defined by induction as follows:*

$$\begin{array}{lcll} \mathcal{A}(x) & := & x & \\ \mathcal{A}(\lambda x.t) & := & \lambda x.\mathcal{A}(t) & \textit{if } x \in \mathrm{FV}(t) \\ \mathcal{A}(\lambda x.t) & := & \lambda x.W_x(\mathcal{A}(t)) & \textit{if } x \notin \mathrm{FV}(t) \\ \mathcal{A}(t\,u) & := & C_\Phi^{\Delta,\Pi}(R_\Delta^\Phi(\mathcal{A}(t))\,R_\Pi^\Phi(\mathcal{A}(u))) & \textit{where } \Phi := \mathrm{FV}(t) \cap \mathrm{FV}(u) \end{array}$$

The encoding adds only the necessary details to ensure linearity – the weakening ensures that a free occurrence of x lies beneath λx and the contraction renames the shared variables of t and u so that the resulting term is linear.

Lengrand provides an encoding of the λ-calculus into Λ_I.

Definition 14 ([9]). *We encode the λ-calculus into λI as follows:*

$$\begin{array}{lcll} i(x) & = & x & \\ i(\lambda x.t) & = & \lambda x.i(t) & x \in \mathrm{FV}(t) \\ i(\lambda x.t) & = & \lambda x.[i(t),x] & x \notin \mathrm{FV}(t) \\ i(t\,u) & = & i(t)\,i(u) & \end{array}$$

This encoding is intended for use in Lengrand's general strategy for proving normalisation properties via simulation in λI. It is the most sensible encoding, only adding anything new when required. The encoding of an abstraction $\lambda x.t$ where $x \notin \mathrm{FV}(t)$ necessarily adds a free occurrence of x, required by the grammar defining Λ_I. However, we now show that i fails in the face of idiocy – and we have not been very sensible in modifying λlxr!

The general proof strategy relies on the relationship $\mathcal{A}(u) \mathrel{\mathcal{J}} i(u)$. Lengrand [9] shows that $\mathcal{A}(u) \mathrel{\mathcal{I}} i(u)$ holds but our use of $\mathcal{J}$ breaks the proof in the case where we may expect it to – in the inductive case involving an abstraction.

Proposition 15. *There exists a λ-term u such that $\mathcal{A}(u)$ is not related by $\mathcal{J}$ to $i(u)$.*

Proof. Assume that $\mathcal{A}(t)\ \mathcal{J}\ i(t)$. Let $u = \lambda x.t$. Whether $x \in \mathrm{FV}(t)$ or not, we can not relate $\mathcal{A}(u)$ to $i(u)$ using $\mathcal{J}$, as the proof trees below suggest. The simplest example is $u = \lambda x.x$ which can be related to $\lambda x.[x,x]$ but not $\lambda x.x = i(u)$.

$$\frac{\mathcal{A}(t)\ \mathcal{J}\ i(t)}{\lambda x.\mathcal{A}(t)\ \mathcal{J}\ \lambda x.[i(t),x]} \qquad \frac{\dfrac{\dfrac{\mathcal{A}(t)\ \mathcal{J}\ i(t)}{\mathcal{A}(t)\ \mathcal{J}\ [i(t),x]}}{W_x(\mathcal{A}(t))\ \mathcal{J}\ [i(t),x]}}{\lambda x.W_x(\mathcal{A}(t))\ \mathcal{J}\ \lambda x.[i(t),x,x]}$$

□

There are two clear ways to address this problem[2] – either redefine $\mathcal{A}$ or i. The latter seems the simplest and more viable and the proof trees in the proposition above suggest the solution – add a redundant x into both i-encodings of abstractions.

Definition 16 ([9]). *We encode the λ-calculus into λI as follows:*

$$\begin{array}{lcll} j(x) & = & x & \\ j(\lambda x.t) & = & \lambda x.[j(t),x] & x \in \mathrm{FV}(t) \\ j(\lambda x.t) & = & \lambda x.[j(t),x,x] & x \notin \mathrm{FV}(t) \\ j(t\,u) & = & j(t)\,j(u) & \end{array}$$

This now allows us to prove our relationship.

Theorem 17. *For any λ-term u, $\mathcal{A}(u)\ \mathcal{J}\ j(u)$.*

Proof. By induction on u. We only treat the case $u = \lambda x.t$ here. By the induction hypothesis, $\mathcal{A}(t)\ \mathcal{J}\ j(t)$. There are two subcases.

- If $x \in \mathrm{FV}(t)$ then $\lambda x.\mathcal{A}(t)\ \mathcal{J}\ \lambda x.[j(t),x] = j(u)$.
- If $x \notin \mathrm{FV}(t)$ then $\lambda x.W_x(\mathcal{A}(t))\ \mathcal{J}\ \lambda x.[j(t),x,x] = j(u)$.

□

This relationship can be depicted as

$$\begin{tikzcd} \lambda\text{-terms} \arrow[d, "\mathcal{A}"] \arrow[dr, "j"] & \\ \lambda\text{blxr-terms} \arrow[r, Rightarrow, "\mathcal{J}"'] & \Lambda_I. \end{tikzcd}$$

So far, we are doing great. We have not only disfigured λlxr but also broken two relationships; one between λlxr-terms and Λ_I, and the other between λ-terms and Λ_I. Is this enough? Unfortunately not as we have to clear up one detail. The proof of PSN for λlxr utilised the fact that i preserved strong normalisation *i.e.* if $t \in \mathrm{SN}_\beta$ then $i(t) \in \mathrm{SN}_{\beta\pi}$. As we are not using i, we need to prove the same proposition for j.

[2] A personal communication from Stéphane Lengrand suggested another solution which seems to lead to a quicker proof of PSN by reusing previous results by Klop. It involves changing both $\mathcal{J}$ and i and is briefly discussed in Section 3.

2.4 The encoding j preserves strong normalisation

We prove that j preserves strong normalisation by adapting Lengrand's proofs with one difference. We omit the typing of λ-abstractions and Π-types which his proofs take into account and concentrate on the special case with no types. The relations $\mathcal{G}$ and $\rightsquigarrow$ defined below are also presented by Lengrand.

In this section, nf^{β} denotes the set of λ-terms which are in β-normal form and $\mathsf{nf}^{\beta\pi}$ denotes the set of λI-terms which are in $\beta\pi$-normal form.

Lemma 18. *For any λ-terms t and u,*

1. $\mathrm{FV}(j(t)) = \mathrm{FV}(t)$
2. $j(t)\{x \setminus j(u)\} = j(t\{x \setminus u\})$

Proof. By induction on t. We treat the cases of abstractions here.

1. Let $t = \lambda x.p$. By the induction hypothesis, $\mathrm{FV}(j(p)) = \mathrm{FV}(p)$. We treat the case where $x \in \mathrm{FV}(P)$. We have

$$\begin{aligned} & \mathrm{FV}(j(\lambda x.p)) \\ = \; & \mathrm{FV}(\lambda x.[j(p), x]) \\ = \; & \mathrm{FV}([j(p), x]) \setminus \{x\} \\ = \; & \mathrm{FV}(p) \setminus \{x\} \\ = \; & \mathrm{FV}(\lambda x.p). \end{aligned}$$

   The case where $x \notin \mathrm{FV}(P)$ is similar.

2. Proof by induction on t. Our alteration to j adds extra occurrences of some variable x which are bound by an abstraction in the term and are hence unaffected by substitution.

$\square$

Definition 19. *Let $\sim_{\beta\pi}$ be the smallest reflexive, transitive relation on Λ_I containing the relation*

$$T \,\mathbf{R}\, U \text{ if } U \longrightarrow_{\beta\pi} T.$$

A term T is $\sim_{\beta\pi}$-related to any term which can $\longrightarrow^{*}_{\beta\pi}$-reduce to it.

Definition 20. *Given a λI term T, the set $T^{\sim_{\beta\pi}}$ is defined as*

$$\{U \mid T \sim_{\beta\pi} V \wedge U \subseteq V\}.$$

Proposition 21. *If $U \longrightarrow_{\beta\pi} T$ then $U^{\sim_{\beta\pi}} \subseteq T^{\sim_{\beta\pi}}$.*

Proposition 22. *If T is strongly normalising and $U \in T^{\sim_{\beta\pi}}$ then U is strongly normalising.*

Proof. By definition, $U \subseteq V \longrightarrow^{+}_{\beta\pi} T$. As T is strongly normalising, V is weakly normalising. By Proposition 5, U is strongly normalising. $\square$

As a diagram, the proposition above reads as follows.

$$\begin{tikzcd} U \subseteq V \arrow[r, two heads, "\beta\pi"'] & T \\ \mathrm{SN} & \mathrm{SN} \arrow[l, Rightarrow, "\text{Proposition 5}"] \end{tikzcd}$$

Definition 23. *The relation $\mathcal{G}$ between λ-terms and λI-terms is given by the following rules where $\overrightarrow{t_k}$ denotes the application $t_1 \dots t_k$:*

$$\frac{\forall k \qquad t_k\ \mathcal{G}\ T_k}{(x\ \overrightarrow{t_k})\ \mathcal{G}\ (x\ \overrightarrow{T_k})}\ \mathcal{G}\mathsf{var} \qquad \frac{}{((\lambda x.t)\ t'\ \overrightarrow{t_k})\ \mathcal{G}\ j((\lambda x.t)\ t'\ \overrightarrow{t_k})}\ \mathcal{G}\beta_1$$

$$\frac{t\ \mathcal{G}\ T \qquad x \in \mathrm{FV}(T)}{\lambda x.t\ \mathcal{G}\ \lambda x.T}\ \mathcal{G}\lambda \qquad \frac{t'\ \mathcal{G}\ T' \qquad x \notin \mathrm{FV}(t)}{((\lambda x.t)\ t'\ \overrightarrow{t_k})\ \mathcal{G}\ (j(\lambda x.t)\ T'\ j(\overrightarrow{t_k}))}\ \mathcal{G}\beta_2$$

$$\frac{t\ \mathcal{G}\ T \qquad N \in \mathrm{SN}_{\beta\pi} \vee N \in T^{\sim_{\beta\pi}}}{t\ \mathcal{G}\ [T, N]}\ \mathcal{G}\mathsf{weak}$$

Again, we have needed to make some changes to the original relation [9]. The changes concern to the $\mathcal{G}$weak rule and we briefly explain why they were necessary in Appendix A. Informally, we allow $\mathcal{G}$weak, a non-deterministic rule, to add 'more' terms than Lengrand's relation. This has implications for the following proofs, most notably Lemma 24.1 where we weaken the consequent from Lengrand's $T \in \mathsf{nf}^{\beta\pi}$ to our $T \in \mathrm{SN}_{\beta\pi}$.

Lemma 24.

1. *If $t \in \mathsf{nf}^{\beta}$ and $t\ \mathcal{G}\ T$, then $T \in \mathrm{SN}_{\beta\pi}$.*
2. *For any λ-term t, $t\ \mathcal{G}\ j(t)$.*

Proof.

1. By induction on t where the $t = (\lambda x.t')\ u\ \overrightarrow{t_k}$ case cannot occur as $t \in \mathsf{nf}^{\beta}$. We first consider the proof tree associated to $t\ \mathcal{G}\ T$ up to a certain point:

   - If $t = x\ \overrightarrow{t_k}$, then one of the last steps of the proof tree associated to $t\ \mathcal{G}\ T$ looks like
   $$\frac{\forall k \qquad t_k\ \mathcal{G}\ T_k}{(x\ \overrightarrow{t_k})\ \mathcal{G}\ (x\ \overrightarrow{T_k}).}$$
   $\overrightarrow{t_k} \subset t$ so $\overrightarrow{t_k} \in \mathsf{nf}^{\beta}$. By the induction hypothesis we have $\overrightarrow{T_k} \in \mathrm{SN}_{\beta\pi}$ and hence $(x\ \overrightarrow{T_k}) \in \mathrm{SN}_{\beta\pi}$.
   - If $t = (\lambda x.u)$, then one of the last steps of the proof tree associated to $t\ \mathcal{G}\ T$ looks like
   $$\frac{u\ \mathcal{G}\ U \qquad x \in \mathrm{FV}(U)}{\lambda x.u\ \mathcal{G}\ \lambda x.U}$$
   $u \subset t$ so $u \in \mathsf{nf}^{\beta}$. By the induction hypothesis we have $U \in \mathrm{SN}_{\beta\pi}$ and hence $\lambda x.U \in \mathrm{SN}_{\beta\pi}$.

   We now consider the remainder of the proof tree associated to $t\ \mathcal{G}\ T$. As t is in β-normal form, only the $\mathcal{G}$weak rule may be used from now on. We induct over the number n of applications of $\mathcal{G}$weak. If $n = 0$ then $T \in \mathrm{SN}_{\beta\pi}$ as shown above. Assume that if $n = k$, $T \in \mathrm{SN}_{\beta\pi}$. Let $n = k + 1$ such that the last step in the tree is
   $$\frac{t\ \mathcal{G}\ T' \qquad N \in \mathrm{SN}_{\beta\pi} \vee N \in T'^{\sim_{\beta\pi}}}{t\ \mathcal{G}\ [T', N]}\ \mathcal{G}\mathsf{weak}$$

where $T = [T', N]$. By the induction hypothesis, $T' \in \mathrm{SN}_{\beta\pi}$ so that $N \in \mathrm{SN}_{\beta\pi}$ by Proposition 22. Therefore, $T \in \mathrm{SN}_{\beta\pi}$.

2. By induction on t:

   - If $t = x\ \overrightarrow{t_k}$, then $t_k\ \mathcal{G}\ j(t_k)$ for all k by the induction hypothesis and we can then apply $\mathcal{G}\mathsf{var}$.
   - If $t = (\lambda x.t')\ u\ \overrightarrow{t_k}$, then $\mathcal{G}\beta_1$ finishes the case.
   - If $t = (\lambda x.u)$, then $u\ \mathcal{G}\ j(u)$ by the induction hypothesis.
     - If $x \in \mathrm{FV}(u)$ then $j(t) = \lambda x.[j(u), x]$.

$$\dfrac{\dfrac{u\ \mathcal{G}\ j(u) \qquad x \in \mathrm{SN}_{\beta\pi}}{u\ \mathcal{G}\ [j(u), x]} \qquad x \in \mathrm{FV}([j(u), x])}{\lambda x.u\ \mathcal{G}\ \lambda x.[j(u), x]}$$

     - If $x \notin \mathrm{FV}(u)$ then $j(t) = \lambda x.[j(u), x, x]$.

$$\dfrac{\dfrac{u\ \mathcal{G}\ j(u) \qquad x \in \mathrm{SN}_{\beta\pi}}{u\ \mathcal{G}\ [j(u), x, x]} \qquad x \in \mathrm{FV}([j(u), x, x])}{\lambda x.u\ \mathcal{G}\ \lambda x.[j(u), x, x]}$$

□

Definition 25. *The reduction relation $\rightsquigarrow$ for λ-terms is defined by the following rules:*

$$\dfrac{t \rightsquigarrow t'}{x\ \overrightarrow{t_k}\ t\ \overrightarrow{p_k} \rightsquigarrow x\ \overrightarrow{t_k}\ t'\ \overrightarrow{p_k}}\ \mathsf{perp\text{-}var} \qquad \dfrac{t \rightsquigarrow t'}{\lambda x.t \rightsquigarrow \lambda x.t'}\ \mathsf{perp}\lambda$$

$$\dfrac{x \in \mathrm{FV}(t) \lor t' \in \mathsf{nf}^{\beta}}{(\lambda x.t)\ t'\ \overrightarrow{t_k} \rightsquigarrow t\{x \setminus t'\}\ \overrightarrow{t_k}}\ \mathsf{perp}\beta_1 \qquad \dfrac{t' \rightsquigarrow t'' \qquad x \notin \mathrm{FV}(t)}{(\lambda x.t)\ t'\ \overrightarrow{t_k} \rightsquigarrow (\lambda x.t)\ t''\ \overrightarrow{t_k}}\ \mathsf{perp}\beta_2$$

Theorem 26. *$\longrightarrow_{\beta\pi}$ strongly simulates $\rightsquigarrow$ through $\mathcal{G}$.*

Proof. We prove that given the diagram

$$\begin{tikzcd} \lambda\text{-terms} \arrow[d, Rightarrow, "\mathcal{G}"] & u \arrow[r, rightsquigarrow] \arrow[d, Rightarrow, "\mathcal{G}"] & u' \arrow[d, Rightarrow, dotted, "\mathcal{G}"] \\ \Lambda_I & U \arrow[r, dotted, "+", "\beta\pi"'] & U', \end{tikzcd}$$

the dotted arrows may always be filled in. We prove by inducting over the structure of u. Figure 4 depicts the various cases. We begin by considering terms U where the last step is not a $\mathcal{G}\mathsf{weak}$ step.

$\mathsf{perp}\beta_1)$ $u = (\lambda x.t)\, t'\, \overrightarrow{t_k} \rightsquigarrow t\{x \setminus t'\}\, \overrightarrow{t_k} = u'$

- $x \in \mathrm{FV}(t)$:

  The final steps of the proof tree for $u \;\mathcal{G}\; U$ must be $\mathcal{G}\beta_1$.Therefore,

$$\begin{array}{rcl} U & = & j(\lambda x.t)\, j(t')\, j(\overrightarrow{t_k}) \\ & = & (\lambda x.[j(t), x])\, j(t')\, j(\overrightarrow{t_k}) \\ & \longrightarrow_\beta & [j(t)\{x \setminus j(t')\}, j(t')]\, j(\overrightarrow{t_k}) \\ & \text{Lemma 18.2} & [j(t\{x \setminus t'\}), j(t')]\, j(\overrightarrow{t_k}) \\ & \longrightarrow_\pi & [j(t\{x \setminus t'\})\, j(\overrightarrow{t_k}), j(t')\,] \\ & = & [j(t\{x \setminus t'\}\, \overrightarrow{t_k}), j(t')\,] \end{array}$$

  By Lemma 24.2,

$$u' = t\{x \setminus t'\}\, \overrightarrow{t_k} \;\mathcal{G}\; j(t\{x \setminus t'\}\, \overrightarrow{t_k}).$$

  As $x \in \mathrm{FV}(t), j(t') \subseteq j(t\{x \setminus t'\}\, \overrightarrow{t_k})$ so we can apply $\mathcal{G}\mathsf{weak}$ to infer $u' \;\mathcal{G}\; [j(t\{x \setminus t'\}\, \overrightarrow{t_k}), j(t')] = U'$.

- $x \notin \mathrm{FV}(t), t' \in \mathsf{nf}^\beta$:

  As $x \notin \mathrm{FV}(t)$, $t' \in \mathsf{nf}^\beta$ and $u' = t\{x \setminus t'\}\, \overrightarrow{t_k} = t\, \overrightarrow{t_k}$. The final steps of the proof tree for $u \;\mathcal{G}\; U$ must be $\mathcal{G}\beta_1$ or $\mathcal{G}\beta_2$. In both cases, $U = \lambda x.[j(t), x, x]\; T'\; j(\overrightarrow{t_k})$ with $t' \;\mathcal{G}\; T'$ (in the former case, $t' \;\mathcal{G}\; j(t') = T'$ by Lemma 24.2). As $t' \in \mathsf{nf}^\beta$, $T' \in \mathrm{SN}_{\beta\pi}$ by Lemma 24.1.

$$\begin{array}{rcl} U & = & \lambda x.[j(t), x, x]\; T'\; j(\overrightarrow{t_k}) \\ & \longrightarrow_\beta, \text{Lemma 18.1} & [j(t), T', T']\; j(\overrightarrow{t_k}) \\ & \longrightarrow^*_\pi & [j(t)\, j(\overrightarrow{t_k}), T', T'\,] \\ & = & [j(t\, \overrightarrow{t_k}), T', T'\,] \end{array}$$

  By Lemma 24.2, $u' = t\, \overrightarrow{t_k} \;\mathcal{G}\; j(t\, \overrightarrow{t_k})$. As $T' \in \mathrm{SN}_{\beta\pi}$, we can apply $\mathcal{G}\mathsf{weak}$ twice to infer $t\, \overrightarrow{t_k} \;\mathcal{G}\; [j(t\, \overrightarrow{t_k}), T', T'\,]$.

$\mathsf{perp}\beta_2)$ $u = (\lambda x.t)\, t'\, \overrightarrow{t_k} \rightsquigarrow (\lambda x.t)\, t''\, \overrightarrow{t_k} = u'$, with $t' \rightsquigarrow t''$ and $x \notin \mathrm{FV}(t)$.

The final steps to prove $u \;\mathcal{G}\; U$ must be $\mathcal{G}\beta_1$ or $\mathcal{G}\beta_2$. In both cases, $U = \lambda x.[j(t), x, x]\; T'\; j(\overrightarrow{t_k})$ with $t' \;\mathcal{G}\; T'$ (in the former case, $t' \;\mathcal{G}\; j(t') = T'$ by Lemma 24.2). By the induction hypothesis, we have

$$\begin{tikzcd} t' \arrow[r, rightsquigarrow] \arrow[d, Rightarrow, "\mathcal{G}"] & t'' \arrow[d, Rightarrow, "\mathcal{G}"] \\ T' \arrow[r, "+"', "\beta\pi"'] & T'' \end{tikzcd}$$

so $U = \lambda x.[j(t), x, x]\; T'\; j(\overrightarrow{t_k}) \longrightarrow^+_{\beta\pi} \lambda x.[j(t), x, x]\; T''\; j(\overrightarrow{t_k})$. We can use rule $\mathcal{G}\beta_2$ to show $u' = (\lambda x.t)\, t''\, \overrightarrow{t_k} \;\mathcal{G}\; [\lambda x.[j(t), x, x]\; T''\; j(\overrightarrow{t_k})]$.

perpλ) $u = \lambda x.t \rightsquigarrow \lambda x.t' = u'$ with $t \rightsquigarrow t'$.

The final steps to prove $u \; \mathcal{G} \; U$ must be $\mathcal{G}\lambda$ so $U = \lambda x.T$ with $t \; \mathcal{G} \; T$. By the induction hypothesis, we have

$$\begin{tikzcd} t \arrow[r, rightsquigarrow] \arrow[d, Rightarrow, "\mathcal{G}"] & t' \arrow[d, Rightarrow, "\mathcal{G}"] \\ T \arrow[r, "+", "\beta\pi"'] & T' \end{tikzcd}$$

so $U = \lambda x.T \longrightarrow^{+}_{\beta\pi} \lambda x.T'$. $x \in \mathrm{FV}(T')$ so we can use rule $\mathcal{G}\lambda$ to show $u' = \lambda x.t' \; \mathcal{G} \; \lambda x.T'$.

perp-var) $u = x \, \overrightarrow{p_k} \, t \, \overrightarrow{q_k} \rightsquigarrow x \, \overrightarrow{p_k} \, t' \, \overrightarrow{q_k} = u'$, with $t \rightsquigarrow t'$.

The final steps to prove $u \; \mathcal{G} \; U$ must be $\mathcal{G}$var so $U = x \, \overrightarrow{P_k} \, T \, \overrightarrow{Q_k}$ with $p_k \; \mathcal{G} \; P_k$, $t \; \mathcal{G} \; T$, and $q_k \; \mathcal{G} \; Q_k$. By the induction hypothesis, there is a term T' such that $t' \; \mathcal{G} \; T'$ and $T \longrightarrow^{+}_{\beta\pi} T'$ as depicted in the last case. Therefore,

$$U = x \, \overrightarrow{P_k} \, T \, \overrightarrow{Q_k} \longrightarrow^{+}_{\beta\pi} x \, \overrightarrow{P_k} \, T' \, \overrightarrow{Q_k}.$$

We can use rule $\mathcal{G}$var to show $u' = x \, \overrightarrow{p_k} \, t' \, \overrightarrow{q_k} \; \mathcal{G} \; x \, \overrightarrow{P_k} \, T' \, \overrightarrow{Q_k}$.

We have only reasoned about the terms U related to u by $\mathcal{G}$ where the last step in the proof tree did not use the $\mathcal{G}$weak rule. The remaining terms can be denoted as $[U, \overrightarrow{N}]$ where the last step of the proof of $u \; \mathcal{G} \; U$ is not a $\mathcal{G}$weak step and the remaining steps which add the term $\overrightarrow{N} = N_1, \ldots, N_m$ are all $\mathcal{G}$weak steps. We induct over m to complete the proof.

If $m = 0$ then the proof follows from the cases above. Assume that the proof holds for $m = k$ with $\overrightarrow{N} = N_1, \ldots, N_k$. From the cases above, this amounts to a proof of the diagram on the left below.

$$\begin{tikzcd} u \arrow[r, rightsquigarrow] \arrow[d, Rightarrow, "\mathcal{G}"] & u' \arrow[d, Rightarrow, "\mathcal{G}"] \\ {[U, \overrightarrow{N}]} \arrow[r, "+", "\beta\pi"'] & {[U', \overrightarrow{N}]} \end{tikzcd} \qquad \begin{tikzcd} u \arrow[r, rightsquigarrow] \arrow[d, Rightarrow, "\mathcal{G}"] & u' \arrow[d, Rightarrow, dotted, "\mathcal{G}"] \\ {[[U, \overrightarrow{N}], N_{k+1}]} \arrow[r, "+", "\beta\pi"'] & {[[U', \overrightarrow{N}], N_{k+1}]} \end{tikzcd}$$

Let $m = k + 1$. We have to complete the diagram on the right above, knowing that $u' \; \mathcal{G} \; [U', \overrightarrow{N}]$. If $N_{k+1} \in \mathrm{SN}_{\beta\pi}$, we apply $\mathcal{G}$weak. Otherwise, we have $N_{k+1} \in [U, \overrightarrow{N}]^{\sim_{\beta\pi}}$. By Proposition 21 and the diagram on the left above, $N_{k+1} \in [U', \overrightarrow{N}]^{\sim_{\beta\pi}}$ and we apply $\mathcal{G}$weak. □

Corollary 27. *If $t \in \mathrm{WN}_{\rightsquigarrow}$ and $t \; \mathcal{G} \; T$ then $T \in \mathrm{WN}_{\beta\pi}$.*

Proof. We prove by induction in $\mathrm{WN}_{\rightsquigarrow}$. Letting Λ denote the set of λ-terms, the induction hypothesis is:

$$(t \in \mathsf{nf}^{\rightsquigarrow}) \lor (\exists u \in \{p \in \Lambda \mid t \rightsquigarrow p\}, \forall U, u \; \mathcal{G} \; U \Rightarrow U \in \mathrm{WN}_{\beta\pi})$$

perpβ_1
$x \in \mathrm{FV}(t)$

$$\begin{tikzcd}
(\lambda x.t)\, t'\, \overrightarrow{t_k} \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & t\{x \setminus t'\}\overrightarrow{t_k} \arrow[d, Rightarrow, "\mathcal{G}"] \\
[\lambda x.[j(t), x]\; j(t')\; j(\overrightarrow{t_k}), \overrightarrow{N}\,] \arrow[r, "+", "\beta\pi"'] & [j(t\{x \setminus t'\}\; \overrightarrow{t_k}), j(t'), \overrightarrow{N}\,]
\end{tikzcd}$$

perpβ_1
$x \notin \mathrm{FV}(t)$
$t'\ \mathcal{G}\ T'$

$$\begin{tikzcd}
(\lambda x.t)\, t'\, \overrightarrow{t_k} \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & t\{x \setminus t'\}\overrightarrow{t_k} \arrow[d, Rightarrow, "\mathcal{G}"] \\
[\lambda x.[j(t), x, x]\;\; T'\; j(\overrightarrow{t_k}), \overrightarrow{N}\,] \arrow[r, "+", "\beta\pi"'] & [j(t\; \overrightarrow{t_k}), T', T', \overrightarrow{N}\,]
\end{tikzcd}$$

perpβ_2
$x \notin \mathrm{FV}(t)$
$t' \rightsquigarrow t''$
$t'\ \mathcal{G}\ T'$

$$\begin{tikzcd}
(\lambda x.t)\, t'\, \overrightarrow{t_k} \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & (\lambda x.t)\, t''\, \overrightarrow{t_k} \arrow[d, Rightarrow, "\mathcal{G}"] \\
[\lambda x.[j(t), x, x]\;\; T'\; j(\overrightarrow{t_k}), \overrightarrow{N}\,] \arrow[r, "+", "\beta\pi"'] & [\lambda x.[j(t), x, x]\;\; T''\; j(\overrightarrow{t_k}), \overrightarrow{N}\,]
\end{tikzcd}$$

perpλ
$t \rightsquigarrow t'$
$t\ \mathcal{G}\ T$

$$\begin{tikzcd}
\lambda x.t \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & \lambda x.t' \arrow[d, Rightarrow, "\mathcal{G}"] \\
[\lambda x.T, \overrightarrow{N}] \arrow[r, "+", "\beta\pi"'] & [\lambda x.T', \overrightarrow{N}]
\end{tikzcd}$$

perp-var
$x \notin \mathrm{FV}(t)$
$t \rightsquigarrow t'$
$t\ \mathcal{G}\ T$

$$\begin{tikzcd}
x\, \overrightarrow{p_k}\, t\, \overrightarrow{q_k} \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & x\, \overrightarrow{p_k}\, t'\, \overrightarrow{q_k} \arrow[d, Rightarrow, "\mathcal{G}"] \\
[x\, \overrightarrow{P_k}\, T\, \overrightarrow{Q_k}, \overrightarrow{N}\,] \arrow[r, "+", "\beta\pi"'] & [x\, \overrightarrow{P_k}\, T'\, \overrightarrow{Q_k}, \overrightarrow{N}\,]
\end{tikzcd}$$

Figure 4: Strong simulation of $\rightsquigarrow$ through $\mathcal{G}$

i.e. either t is in $\rightsquigarrow$-normal form or there exists a one-step $\rightsquigarrow$-reduct u of t such that the proposition holds (all $\mathcal{G}$-related terms of u are $\longrightarrow_{\beta\pi}$-weakly normalising.)

If $t \in \mathsf{nf}^{\rightsquigarrow}$, then $T \in \mathrm{SN}_{\beta\pi} \subseteq \mathrm{WN}_{\beta\pi}$ by Lemma 24.1.

If $\exists u \in \{p \in \Lambda \,|\, t \rightsquigarrow p\}, \forall U, u\ \mathcal{G}\ U \Rightarrow U \in \mathrm{WN}_{\beta\pi}$, then Theorem 26 gives us a specific T' such that

$$\begin{tikzcd}
t \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & u \arrow[d, Rightarrow, "\mathcal{G}"] \\
T \arrow[r, "+", "\beta\pi"'] & T'.
\end{tikzcd}$$

According to the induction hypothesis, $T' \in \mathrm{WN}_{\beta\pi}$, and so $T \in \mathrm{WN}_{\beta\pi}$. $\square$

Corollary 28. $j(\mathrm{SN}_\beta) \subseteq \mathrm{WN}_{\beta\pi}$.

Proof. $\mathrm{SN}_\beta \subseteq \mathrm{SN}_{\leadsto} \subseteq \mathrm{WN}_{\leadsto}$. By Lemma 24.2, $\forall t \in \mathrm{SN}_\beta, t \; \mathcal{G} \; j(t)$. By the previous corollary, $j(t) \in \mathrm{WN}_{\beta\pi}$. ☐

Theorem 29 (Nederpelt[11]). $\mathrm{WN}_{\beta\pi} \subseteq \mathrm{SN}_{\beta\pi}$.

Corollary 30. *For any* λ*-term* t*, if* $t \in \mathrm{SN}_\beta$*, then* $j(t) \in \mathrm{SN}_{\beta\pi}$*.*

Proof. By Corollary 28 and Theorem 29. ☐

2.5 Proof of PSN

Corollary 31 (PSN). *For any* λ*-term* t*, if* $t \in \mathrm{SN}_\beta$*, then* $\mathcal{A}(t) \in \mathrm{SN}_{\lambda\mathrm{blxr}}$*.*

Proof. If $t \in \mathrm{SN}_\beta$ then $j(t) \in \mathrm{SN}_{\beta\pi}$ by Corollary 30. As $\mathcal{A}(t) \; \mathcal{J} \; j(t)$ by Theorem 17, $\mathcal{A}(t) \in \mathrm{SN}_{\lambda\mathrm{blxr}}$ by Corollary 12. ☐

3 Simplification of the proof

On showing him this work, Stéphane Lengrand had another idea to fix the problem of Section 2.3 that $\mathcal{A}(u) \; \mathcal{J} \; i(u)$ does not hold. It consisted of changing the $\mathcal{J}$ relation in the abstraction case and using Klop's encoding [7, Definition 8.11]:

$$\begin{array}{lcl} \imath(x) & = & x \\ \imath(\lambda x.t) & = & \lambda x.[\imath(t), x] \\ \imath(t\,u) & = & \imath(t)\,\imath(u). \end{array}$$

This approach may allow us to use previous results by Klop to complete the proof and should yield a simpler solution.

4 Summary

In summary, we have proved PSN for λblxr using the proof for λlxr with a few modifications due to the replacement of $\longrightarrow_B$ with $\longrightarrow_{Bs}$. The modifications we made were:

1. Altering the relation $\mathcal{I}$ so that $\longrightarrow_{\lambda\mathrm{blxr}}$ was simulated by $\longrightarrow_{\beta\pi}$ through the new relation $\mathcal{J}$.

2. Altering the encoding i so that $\mathcal{A}(u)$ was related by $\mathcal{J}$ to the new encoding j.

3. Proving that j preserved strong normalisation just as Lengrand proved that i preserved strong normalisation. This required altering the relation $\mathcal{G}$ in the $\mathcal{G}\mathsf{weak}$ case and weakening the remaining propositions.

The alterations to $\mathcal{I}$ and i consisted in adding in some redundancy and the alteration to $\mathcal{G}\mathsf{weak}$ was made to accomodate for this. This redundancy took the form of tacking on (via the "memory operator") a free variable x to a λI-term which already contains a free occurrence of x. This is analogous to the redundancy in the $\longrightarrow_{Bs}$ rule which creates a second, identical explicit substitution in its firing.

5 Acknowledgements

Stéphane Lengrand looked over a near-complete version of this proof during the HOR 2006 workshop which was extremely helpful. Jan Willem Klop was very helpful in his responses to my questions about his work which is instrumental to both the original proofs, our Definition 20, and our version of the $\mathcal{G}\mathsf{weak}$ rule.

References

[1] Alonzo Church. *The Calculi of Lambda Conversion.* Princeton University Press, Princeton, NJ, 1941. Reprinted 1963 by University Microfilms, Ann Arbor, MI.

[2] René David and Bruno Guillaume. A lambda-calculus with explicit weakening and explicit substitution. *Mathematical Structures in Computer Science*, 11(1):169–206, 2001.

[3] Maribel Fernández and Ian Mackie. Closed reductions in the lambda-calculus. In Jörg Flum and Mario Rodríguez-Artalejo, editors, *Computer Science Logic*, volume 1683 of *Lecture Notes in Computer Science*, pages 220–234. Springer, 1999.

[4] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[5] Delia Kesner and Stéphane Lengrand. Explicit operators for lambda-calculus. Available at "*http://www.pps.jussieu.fr/∼kesner/papers/*".

[6] Delia Kesner and Stéphane Lengrand. Extending the explicit substitution paradigm. In Jürgen Giesl, editor, *RTA*, volume 3467 of *Lecture Notes in Computer Science*, pages 407–422. Springer, 2005.

[7] Jan Willem Klop. *Combinatory Reduction Systems. PhD Thesis*, volume 127 of *Mathematical Centre Tracts*. CWI, Amsterdam, 1980.

[8] Yves Lafont. Interaction nets. In *POPL '90: Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 95–108, New York, NY, USA, 1990. ACM Press.

[9] Stéphane Lengrand. Induction principles as the foundation of the theory of normalisation: Concepts and techniques. Technical report, PPS laboratory, Université Paris 7, March 2005. available at `http://hal.ccsd.cnrs.fr/ccsd-00004358`.

[10] Robin Milner. Local bigraphs, confluence and λ-calculus (draft), 2004.

[11] R. P. Nederpelt. *Strong normalization in a typed lambda calculus with lambda structured types.* PhD thesis, Eindhoven University of Technology, 1973.

[12] Shane Ó Conchúir. Λ_{sub} as an explicit substitution calculus. Technical report, IT-Universitetet, København, submitted September 2006.

A Change to the $\mathcal{G}$ relation

The original relation $\mathcal{G}$ [9] was defined as in Definition 23 except that $\mathcal{G}\mathsf{weak}$ was defined as below.

$$\frac{t\ \mathcal{G}\ T \qquad N \in \mathsf{nf}^{\beta\pi}}{t\ \mathcal{G}\ [T, N]}\ \mathcal{G}\mathsf{weak}$$

This relation was not sufficient for our choice of j. One problem lay in Theorem 26 for the $\mathsf{perp}\beta_1$ case where $x \in \mathrm{FV}(t)$. In this case we have

$$u = (\lambda x.t)t'\overrightarrow{t_k} \rightsquigarrow t\{x \setminus t'\}\overrightarrow{t_k} = u'.$$

We have $u\ \mathcal{G}\ [(\lambda x.[j(t), x])j(t')j(\overrightarrow{t_k}), \overrightarrow{N}] \longrightarrow_{\beta\pi} [j(t\{x \setminus t'\}\overrightarrow{t_k}), j(t'), \overrightarrow{N}]$ and we can show that $u'\ \mathcal{G}\ j(t\{x \setminus t'\}\overrightarrow{t_k})$. The problem is the next step. We need to add $j(t')$ and $\overrightarrow{N}$ onto this last term to complete the proof. This requires applications of the $\mathcal{G}\mathsf{weak}$ rule but $j(t') \notin \mathsf{nf}^{\beta\pi}$ in general and we cannot proceed. As a concrete example, take the diagram below. Although $\Omega\ \mathcal{G}\ j(\Omega)$, we cannot apply $\mathcal{G}\mathsf{weak}$ to complete the diagram as $j(\Omega) \notin \mathsf{nf}^{\beta\pi}$.

$$\begin{tikzcd}
(\lambda x.x)\Omega \arrow[r, squiggly] \arrow[d, Rightarrow, "\mathcal{G}"] & \Omega \arrow[d, dotted, "\times\mathcal{G}"] \\
(\lambda x.[x,x])j(\Omega) \arrow[r, "+", "\beta\pi"'] & {[j(\Omega), j(\Omega)]}
\end{tikzcd}$$

Essentially, the simulation of Theorem 26 does not work due to the redundant x introduced in the encoding of $j(\lambda x.t)$ where $x \in t$ – the redundant x creates duplicates of arguments through β-reductions. These arguments may not be in normal form. To accomodate for this, our first attempt was to change $\mathcal{G}\mathsf{weak}$ to the following where we weaken the right-hand side condition.

$$\frac{t\ \mathcal{G}\ T \qquad N \in \mathsf{nf}^{\beta\pi} \vee N \subseteq T}{t\ \mathcal{G}\ [T, N]}\ \mathcal{G}\mathsf{weak}$$

Now, given $u\ \mathcal{G}\ [(\lambda x.[j(t), x])j(t')j(\overrightarrow{t_k}), \overrightarrow{N}] = U \longrightarrow_{\beta\pi} [j(t\{x \setminus t'\}\overrightarrow{t_k}), j(t'), \overrightarrow{N}]$, we can show that $u'\ \mathcal{G}\ [j(t\{x \setminus t'\}\overrightarrow{t_k}), j(t')] = U'$. Unfortunately, this simple solution has a problem – the term $\overrightarrow{N}$ may be composed of subterms of U but these terms may not be subterms of U' and we cannot complete the case. In particular, a counterexample may be found by considering the λ abstraction in U. For example, let $u = (\lambda x.x\Omega)x \rightsquigarrow x\Omega = u'$. We can prove $u\ \mathcal{G}\ [(\lambda x.[xj(\Omega), x])x, (\lambda x.[xj(\Omega), x])] \longrightarrow_{\beta\pi} [xj(\Omega), x, \lambda x.[xj(\Omega), x]]$ but cannot prove that u' is $\mathcal{G}$-related to this last term.

These problems led us to the current definition of $\mathcal{G}\mathsf{weak}$ which can be seen as a generalisation of the last definition above as it allows subterms of ancestors to be added on to a term. The fact that λI is uniformly normalising (by Proposition 5) was important for our current choice of $\mathcal{G}\mathsf{weak}$ in order to prove Lemma 24.1.