A calculus of multiary sequent terms

JOSE ESPÍRITO SANTO

and

LUIS PINTO

Universidade do Minho

APPENDIX

A. PROOF TRANSFORMATIONS

In this appendix we show the proof transformations associated to each reduction rule and permutative conversion, as required by the proofs of Theorems 2.7 and 2.9 on subject reduction and subject permutation.

Preliminaries. In defining a proof transformation, we use the admissible rules proved in Propositions 2.4 and 2.5. This should be understood as indicating that the transformation also comprises the obtention of the conclusion of the admissible rule by the transformation of its premisses, according to the process that is associated to the admissibility of the rule, and that constitutes the content of the proof of that admissibility.

The content of the proof of Proposition 2.4, concerned with the admissibility of the forms (8) of mid-cut, is a certain transformation on derivations: given a derivation $D_1$ of $\Gamma \vdash t : A$, and given another derivation $D_2$ of $x : A, \Gamma \vdash v : B$ (resp. of $x : A, \Gamma \vdash v : B$), there is a derivation $D_3$ of $\Gamma \vdash s(t, x, v) : B$ (resp. of $\Gamma \vdash s(t, x, v) : B$), obtained from $D_1$ and $D_2$ by an obvious (but tedious to define) process that we describe as the complete permutation, along the subderivation of the right premiss, of the implicit cut of the form of the right (resp. right) figure of (8) with premisses derived by $D_1$ and $D_2$.

The content of the proof of Proposition 2.5, concerned with the admissibility of the form (9) of head-cut, is another transformation of derivations. Given derivations $D_1$ of $\Gamma ; C \vdash l : A_1$ and $D_2$ of $\Gamma \vdash u' : A_1$ and $D_3$ of $\Gamma ; A_2 \vdash l' : B$, there is a derivation $D_4$ of $\Gamma ; C \vdash a(l, u', l') : B$, obtained from $D_1$, $D_2$ and $D_3$ by an obvious process that we describe as the complete permutation, along the subderivation of the left premiss, of the implicit cut of the form (10) with premisses derived by $D_1$, $D_2$ and $D_3$.

For readability, in proof transformations we omit contexts as well as term and list annotations from sequents; we use terms and lists for naming derivations instead,
and we choose these names exactly as in the definitions of reduction rules and permutative conversions (recall Definition 2.6 and Subsection 2.4). In addition, we do not make explicit the uses of the weakening and strengthening rules.

Reduction rules.

Case $\beta_1$. The LHS corresponds to a derivation of the form:

$$
\vdash A \supset B \quad \text{Right} \quad \vdash A \vdash B \quad Ax \quad \vdash y : B \vdash C \quad \text{lm-Left} \quad \vdash C
$$

In such cut inference the cut formula is main in both premisses. This determines a key step in cut-elimination and the elimination of this cut produces two mid-cuts:

$$
\vdash A \quad \vdash B \quad \text{mid-cut} \quad \vdash y : B \vdash C \quad \text{mid-cut}
$$

Case $\beta_2$. The derivation corresponding to the LHS has the form

$$
\vdash A \vdash B_1 \supset B_2 \quad \text{Right} \quad \vdash A \vdash B_1 \supset B_2 \quad \text{lm-Left} \quad \vdash B_1 \supset B_2 \vdash D
$$

Again, we are in the presence of a key step in cut-elimination. However, in this case one of the cuts generated assumes the particular form corresponding to a gm-elimination:

$$
\vdash A \quad \vdash B_1 \supset B_2 \quad \text{mid-cut} \quad \vdash B_1 \supset B_2 \vdash D \quad \text{lm-Left} \quad \vdash D
$$

Case $\pi$. The LHS corresponds to a derivation of the form

$$
\vdash A \supset B \quad \vdash A \vdash B \quad x : C \vdash D_1 \supset D_2 \quad h\text{-cut} \quad \vdash D_1 \supset D_2 \vdash F \quad h\text{-cut}
$$

where (a) and (b) are \textit{lm-Left} inferences. Since the left cut formula of the outer cut is not main, this cut is left-permutable. The RHS of $\pi$ results by permuting the
outer cut above the other cut and inference (a):

\[
\begin{array}{cccccc}
  u & l & v & u' & l' & v' \\
  t & x:C \vdash D_1 & x:C; D_2 \vdash E & y: E, x:C \vdash F & \hline \\
  \vdash A \vdash B & \vdash :A \vdash B \vdash E & \hline \\
  \vdash F & \hline \\
\end{array}
\]

Case \( \mu \). The LHS corresponds to a derivation of the form:

\[
\begin{array}{cccccc}
  u & l & v & u' & l' & v \\
  t & \vdash A \vdash B \vdash C_1 \vdash C_2 & \vdash x:C_1 \vdash C_2 \vdash E & \hline \\
  \vdash A \vdash B \vdash E & \hline \\
  \vdash F & \hline \\
\end{array}
\]

Notice that the variable \( x \) does not occur in \( u', l', v \) and the inner \( \text{gm-application} \) is an \( m-\text{Left} \) introduction whose main formula is active in the next inference. So formula \( C_1 \supset C_2 \), instead of being introduced by an \( m-\text{Left} \) inference, could have been introduced in a linear fashion by a \( \text{Lft} \) inference:

\[
\begin{array}{cccccc}
  u & l & v & u' & l' \\
  t & \vdash A \vdash B \vdash C_1 \vdash C_2 & \vdash :B \vdash D & \vdash y: E, x:C \vdash F & \hline \\
  \vdash A \vdash B \vdash E & \hline \\
  \vdash F & \hline \\
\end{array}
\]

This construction requires subderivations indicated by \( u', l' \) and \( v \) to be strengthened by erasure of declaration \( x:C_1 \supset C_2 \).

**Permutative conversions.**

Case \( p_1 \). The LHS corresponds to a derivation of the form:

\[
\begin{array}{cccccc}
  u & l & y: D \vdash A & x: C; y: D \vdash C & \hline \\
  y: D \vdash A \vdash B & \vdash :y: D, A \vdash B \vdash D & \hline \\
  y: D \vdash D & \hline \\
\end{array}
\]

The endsequent can in this case simply be obtained by:

\[
\begin{array}{cccccc}
  y: D \vdash D & \hline \\
\end{array}
\]

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.
Case $p_2$. The LHS corresponds to a derivation of the form:

\[
\begin{array}{c}
\frac{\vdash A \supset B}{\vdash A \supset D_1 \supset D_2} \text{lm-Left} \\
\frac{x:C, y:D_1 \vdash D_2}{\vdash A \supset B} \quad \text{Right}
\end{array}
\]

If $x \in v$, the effect of the transformation is to permute the right inference down past the two inferences \text{lm-Left} and \text{h-cut}:

\[
\begin{array}{c}
\frac{y:D_1 \vdash D_2}{\vdash A \supset D_2} \quad \text{Right} \\
\frac{x:C, y:D_1 \vdash D_2}{\vdash A \supset B} \quad \text{lm-Left}
\end{array}
\]

If $x \notin v$, admissibility of strengthening justifies the following derivation:

\[
\begin{array}{c}
\frac{y:D_1 \vdash D_2}{\vdash A \supset D_2} \quad \text{Right}
\end{array}
\]

Consider now the special case of permutation $p_2$ where $x \in v$ and where $t$ is a variable, say $z$. Its RHS corresponds to

\[
\begin{array}{c}
\frac{z:A \supset B \vdash A}{\vdash A \supset B} \quad \text{Right} \\
\frac{x:C, z:A \supset B \vdash D_2}{\vdash A \supset B} \quad \text{m-Left}
\end{array}
\]

The transformation permutes the right inference below a multiary left inference:

\[
\begin{array}{c}
\frac{y:D_1, z:A \supset B \vdash A}{\vdash A \supset B} \quad \text{Right} \\
\frac{x:C, y:D_1, z:A \supset B \vdash D_2}{\vdash A \supset B} \quad \text{m-Left}
\end{array}
\]

This transformation relates to permutative conversion (5) of [Schwichtenberg 1999] and, in the unary case ($l = []$), to the permutation of an implication right inference below an ordinary implication left inference.

Case $p_3$. The LHS corresponds to a derivation of the form:
A calculus of multiary sequent terms

\[ t_1 \vdash A_1 \supset B_1 \quad D_1 \quad D \quad (23) \]

where \( D_1 \) is

\[ t_2 \vdash A_2 \supset B_2 \quad D_3 \quad D_4 \quad (24) \]

We illustrate what happens if \( x \in t_2 \) and \( x \in u_2 \). (If this is not the case, we have simplified situations.) The corresponding RHS is

\[ v \]

\[ \vdash D_2 \quad t \quad D_3 \quad D_4 \quad (25) \]

where \( D_2 \) is

\[ \vdash A_2 \supset B_2 \quad t \quad (26) \]

(26)

(The admissibility of this rule is an easy induction on \( l' \).)

The role of \( p_3 \) is to permute the block of two inferences \( \text{lm-Left} \) and \( \text{h-cut} \) down past the block of two inferences \( \text{lm-Left} \) and \( \text{h-cut} \), swapping in particular the relative order of the head-cuts on \( A_1 \supset B_1 \) and on \( A_2 \supset B_2 \). In order to perform the permutation, block (23) needs to be propagated to the derivations of the premises of the block (24) represented by \( t_2, u_2, l_2 \) (recall \( x \not\in v \)). In the first two cases it suffices to add block (23) at the end of the derivations corresponding to \( t_2, u_2 \), replacing \( D \) by \( A_2 \supset B_2 \) or \( A_2 \) respectively. In the last case recall that the derivation corresponding to \( l_2 \) is a tower of \( \text{Lft} \) inferences. Here propagation is achieved thus: for each of these \( \text{Lft} \) inferences, add the block (23) at the end of the
derivation of its minor premise, whenever \( x \) occurs in the term representing this
derivation; this is the proof transformation associated with the admissibility of rule
(26).

Consider now the particular case of \( p_3 \) where \( t_1 \) and \( t_2 \) are variables, say \( z_1 \) and
\( z_2 \). Its LHS corresponds to two multiary left inferences introducing \( z_1 : A_1 \supset B_1 \)
and \( z_2 : A_2 \supset B_2 \):

\[
\begin{array}{c}
\frac{u_1 \quad l_1 \quad u_2 \quad l_2 \quad v}{\Gamma \vdash A_1 \quad \Gamma; B_1 \vdash C_1} \quad \frac{\Gamma, x : C_1 \vdash A_2}{\Gamma, x : C_2 \vdash D} \quad m \text{-} Left
\end{array}
\]

where \( \Gamma = \{ z_1 : A_1 \supset B_1, z_2 : A_2 \supset B_2 \} \). In this case, and assuming \( x \in u_2 \), the
transformation on derivations associated with \( p_3 \), as defined before, can be thought
of as the permutation of the two multiary left inferences, resulting in

\[
\begin{array}{c}
\frac{u_1 \quad l_1 \quad u_2 \quad \vdash A_1 \quad \Gamma; B_1 \vdash C_1 \quad \Gamma, x : C_1 \vdash A_2 \quad \vdash D}{\vdash D} \quad m \text{-} Left
\end{array}
\]

as long as derivation \( D_2 \) of (25) is replaced by an axiom (because declaration \( z_2 : A_2 \supset B_2 \) is in the context) so that 25 is an \( m \) \text{-} Left inference.

Case \( q \). The LHS corresponds to a derivation of the form

\[
\begin{array}{c}
\frac{\vdash A \supset (B_1 \supset B_2) \quad \vdash B_1 \supset B_2 \vdash C \quad \vdash A \supset (B_1 \supset B_2) \vdash D}{\vdash D}
\end{array}
\]

and the RHS corresponds to a derivation of the form

\[
\begin{array}{c}
\frac{\vdash B_1 \supset B_2 \vdash C \quad x : C \vdash D \quad \vdash D}{\vdash D}
\end{array}
\]

In contrast to \( p_2 \) and \( p_3 \), the proof transformation associated to \( (q) \) does not
have the flavour of permuting inferences past other inferences. This transformation

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.
forces the main formula \( B_1 \supset B_2 \) of a \textit{Lft} inference to be the main formula of a linear multitype left inference. As a consequence, \( B_1 \supset B_2 \) also becomes the cut-formula of a new head-cut, the right subderivation \( D \) of which is another head-cut, the descendant of the original head-cut with cut-formula \( A \supset (B_1 \supset B_2) \). The middle premise of the displayed \textit{lm-Left} inference of \( D \) has no instance of \textit{Lft}.

\section{RELATIONSHIP WITH \( \lambda \)-CALCULUS}

Herbelin’s \( \lambda \)-calculus (and its type system, the sequent calculus \( LJT \)) [Herbelin 1995] was important for the present paper because we recognised in \( \lambda \) syntactic solutions for the implementation of multarity. \( \lambda \) continues to be used in literature as a presentation of the intuitionistic sequent calculus equipped with a computational interpretation, so a natural question is: what are the differences and relative advantages between \( \lambda \) and \( \lambda \textit{Jm} \)?

Roughly speaking, the difference between the two systems boils down to this: \( \lambda \) has a restricted form of (primitive) left introduction, whereas \( \lambda \textit{Jm} \) has a restricted form of (primitive) cut. In \( \lambda \), left introduction corresponds to \( u :: l \), while more general forms are derivable with the help of cut; in \( \lambda \textit{Jm} \), in addition to \( u :: l \), one has the particular forms of \textit{gm}-application \( x(u, (y)v) \) and \( x(u, l, (y)v) \) that correspond to ordinary and multitype left introduction. On the other hand, in \( \lambda \textit{Jm} \) \textit{gm}-application corresponds to a cut with right cut-formula main in a left introduction (recall figure (6)), while the unconstrained cut is only admissible (as witnessed by the admissibility of the typing rule for substitution - Proposition 2.4); in \( \lambda \) the general form of cut is primitive, and that is why \( \lambda \) has an explicit substitution construction.

We now give some elements of a more formal comparison between \( \lambda \) and \( \lambda \textit{Jm} \).

Expressions of Herbelin’s \( \lambda \)-calculus are, as those of \( \lambda \textit{Jm} \), separated into terms and lists, and are given by:

\[
\begin{align*}
t, u, v & ::= y[l] \mid \lambda x.t \mid tl \mid v\{x := t\} \\
l & ::= [] \mid u :: l
\end{align*}
\]

We are omitting two list constructors for simplicity. In addition to \( \lambda \)-abstraction, the term constructors are \( y[l] \) (dereliction), \( tl \) (head-cut) and \( v\{x := t\} \) (mid-cut), with typing rules

\[
\begin{align*}
y & : A, \Gamma; A \vdash l : B \\
\text{Der}
\end{align*}
\]

\[
\begin{align*}
\Gamma \vdash t : A & \quad \Gamma; A \vdash l : B \\
\Gamma \vdash tl : B & \quad \text{h-cut} \\
\Gamma \vdash t : A & \quad x : A, \Gamma \vdash v : B \\
\Gamma \vdash v\{x := t\} : B & \quad \text{m-cut}
\end{align*}
\]

In \( \lambda \) there are reduction rules to eliminate both forms of cuts. Mid-cuts are explicit substitutions and the reduction rules to eliminate them correspond to steps in the execution of explicit substitution.

We consider the question of mapping \( \lambda \) into \( \lambda \textit{Jm} \). Insofar lists are restricted to the forms \( [] \) and \( u :: l \), the following suffices for a type preserving mapping (whose range is actually contained in \( \lambda \textit{m} \)):

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.
The cut-free fragment of $\lambda$ is, of course, the fragment where constructors $\mathit{tl}$ and $v\{x := t\}$ are omitted. The interpretation of dereliction given by the first two clauses above establishes a bijection between cut-free $\lambda$-terms and the normal forms of $\lambda^m$ as observed in [Espírito Santo 2002a]. The interpretation of mid-cut by meta-substitution means that the reduction steps for the explicit execution of substitution in $\lambda$ are mapped to equations in the target.

Let us now turn into the question of mapping $\lambda^m$ into $\lambda$. There is an interpretation determined by the following transformations:

\begin{align*}
   y[] & \leadsto y \\
   y'(u :: l) & \leadsto y(u, l) \\
   v\{x := t\} & \leadsto s(t, x, v) \\
   t[] & \leadsto t \\
   t(u :: l) & \leadsto t(u, l)
\end{align*}

Thus $\text{gm}$-application is interpreted as explicit substitution. The same interpretation was given in [Schwichtenberg 1999] to $v\{y, u :: l\}$, i.e. $y(u, l, (x)v)$.

In terms of inference rules, (28) corresponds to interpreting $\text{gm}$-eliminations as the following combination of inferences in $\lambda$:

\[
\frac{
\begin{array}{c}
\Gamma \vdash t: A \\
\Gamma; B \vdash l: C
\end{array}
}{
\Gamma; A \supset B \vdash u :: l: C \\
\hline
\Gamma \vdash t(u :: l): C
\}
\]

Thus should be compared with (6): instead of a linear multiary left inference, one has a $\text{Lft}$-inference plus a mid-cut.

Interpretation (28) is related to the translation of $LJ$ left inferences into inferences of $\lambda$ given in [Herbelin 1995]. Writing unary left inferences by means of $\lambda^m$-terms, Herbelin’s translation reads

\[
y(u, (x)v) \leadsto v\{x := y[u]\}
\]

and is readily extended to multiary left inferences thus:

\[
y(u, l, (x)v) \leadsto v\{x := y'(u :: l)\}
\]

In the case $t = y$, (31) is a slight improvement over (28) in the sense that

\[
y(u, l, (x)v) \leadsto v\{x := y'(u :: l)\}
\]

\[
\Rightarrow v\{x := y'(u :: l)\}
\]
where (32) is by (27) and (28), and (33) is a reduction allowed in $\lambda$. The interpretation (28) shows that $\lambda$, or rather its type system, proves the same logical sequents as the type system of $\lambda J_m$. But there are problems with interpretation (28) as a mapping between two structures, problems which are certainly shared to a large extent by Herbelin’s interpretation (30) of $L J$. The cut-free fragment of $\lambda$ is permutation-free [Dyckhoff and Pinto 1999] and, accordingly, is small and has no notion of permutative conversion. So it is not surprising that interpretation (28) does not preserve cut-freeness, and maps permutative conversion steps, at best, to cut-elimination steps (but the simulation of $p_3$-permutations poses problems, as it would require the propagation of a substitution inside another substitution, a feature not available in $\lambda$). Finally, also the simulation of cut-elimination steps is problematic. For instance, a $\pi$-step is mapped to

$$v'\{y := v\{x := t(u :: l)\}\{u' :: l'\}\} \rightarrow v'\{y := v\{u' :: l'\}\}\{x := t(u :: l)\},$$

a step for enlarging the scope of substitution $\{x := t(u :: l)\}$ not reproducible by the cut-elimination rules of $\lambda$.

C. RESULTS ON PERMUTATIVE CONVERSIONS

In this appendix we present, in the first subsection, the proofs of the main results on permutative conversions, established in Section 4.1 in terms of mapping $\phi$; and detail, in the second subsection, analogous results for mappings $p$, $q$, $p^m$ and $qJ$.

C.1 Proofs of main results

Firstly we introduce some basic facts about $\phi$ used throughout.

**Lemma C.1.** For all $t, u, v, v_1, v_2 \in T J_m$, $l \in L J_m$:

1. $\phi(t(u)) = \phi(t)(\phi(u))$;
2. $\phi(t(u, l, (x)v)) = s(\phi(t(u, l)), x, \phi(v)) = \phi(t[u, l, (x)v]$;
3. $\phi(s(t, x, v)) = s(\phi(t), x, \phi(v))$ and
   $\phi'(s(\phi(t), x, \phi(v_1)), s(\phi(t), x, \phi(v_2)), s(t, x, l, y, s(\phi(t), x, \phi(v)))) = s(\phi(t), x, \phi(v_1, v_2, l, (y)v))$.

**Proof.** Part 1 is by a simple calculation:

$$\phi(t(u)) = \phi'(\phi(t), \phi(u), [], x, \phi(x)) = s(\phi(t)(\phi(u)), x, x) = \phi(t)(\phi(u)).$$

The first equality of 2 follows by routine induction on $l$ and the second equality follows then easily from the first. The two conjuncts of 3 are proved together, by simultaneous induction on $v$ and $l$. $\square$

Now we address two key properties in establishing the permutability theorem for mapping $\phi$. These properties assert that permutation reduction is invariant under $\phi$ (Proposition C.2) and that each term can be reduced to its $\phi$-image using solely permutations (Proposition C.4).

**Proposition C.2.** If $t \rightarrow^*_{pq} u$ then $\phi(t) = \phi(u)$, for all $t, u \in T J_m$.
Proof. The proof follows by induction on the relation $\rightarrow_{pq}^*$. Below we consider the base cases corresponding to the various permutations.

Case $p_1$. 
\[ \phi(t(u, (x)y)) \]
\[ = s(\phi(t(u, l), x, \phi(y))) \quad \text{(Lemma C.1.2)} \]
\[ = \phi(y) \quad x \neq y \]

Case $p_2$. 
\[ \phi(t(u, l, (x)\lambda y.v)) \]
\[ = s(\phi(t(u, l), x, \phi(y)) \quad \text{(Lemma C.1.2)} \]
\[ = \lambda y.s(\phi(t(u, l), x, \phi(v)) \]
\[ = \lambda y.\phi(t(u, l, (x)v)) \quad \text{(Lemma C.1.2)} \]
\[ = \phi(\lambda y.t(u, l, (x)v)) \]

Case $p_3$. 
\[ \phi(t_1(u_1, l_1, (x)t_2(u_2, l_2, (y)v))) \]
\[ = s(\phi(t_1(u_1, l_1), x, \phi(t_2(u_2, l_2, (y)v))) \quad \text{(Lemma C.1.2)} \]
\[ = \phi'(s(\phi(t_1(u_1, l_1), x, \phi(t_2(u_2, l_2, (y)v)), s(\phi(t_1(u_1, l_1), x, \phi(t_2(u_2, l_2, (y)v))) \quad \text{(Lemma C.1.2 and fact)} \]
\[ \phi'(t_1(u_1, l_1, (x)t_2(u_2, l_2, (y)v), y, \phi(v)) \quad \text{(IH)} \]
\[ \phi(t_1(u_1, l_1, (x)t_2(u_2, l_2, (y)v))) \quad \text{(Lemma C.1.2)} \]

Case $q$. 
\[ \phi(t(u, v::l, (x)v')) \]
\[ = \phi'(\phi(t)(\phi(u)), \phi(v), l, x, \phi(v')) \]
\[ = \phi(t(u)(v, l, (x)v')) \quad \text{(Lemma C.1.1)} \]

Proposition C.4 is established with the help of the following auxiliary results.

Lemma C.3. For all $t, u, v \in T^m, l \in L^m$, $t\|u, l, (x)v\| \Rightarrow_{p}^* s(t(u, l), x, v)$.

Proof. Proved together with the fact, for all $t, u \in T^m, l, l_0 \in L^m$, 
\[ t\|u, l, (x)l_0\| \Rightarrow_{p}^* s(t(u, l), x, l_0), \]
by induction on $v$ and $l_0$. We show the cases relative to $v$.

Observe that if $x \notin v$, the LHS and the RHS are both equal to $v$. Below we assume $x \in v$.

Case $v = x$. 
\[ t\|u, l, (x)x\| = t(u, l) = s(t(u, l), x, x) \]

Case $v = \lambda y.v_0$. 
\[ t\|u, l, (x)\lambda y.v_0\| = t(u, l, (x)\lambda y.v_0) \quad \text{(IH)} \]
\[ = s(t(u, l), x, \lambda y.v_0) \]

Case $v = t_1(u_1, l_1)$. 
\[ t\|u, l, (x)t_1(u_1, l_1)\| = t(u, l, (x)t_1(u_1, l_1)) \quad \text{(IH)} \]
\[ = s(t(u, l), x, l_1(u_1, l_1)) \]
\[ \Rightarrow_{p}^* s(t(u, l), x, l_1(u_1, l_1)) \]
\[ \Rightarrow_{p}^* s(t(u, l), x, l_1(u_1, l_1)) \]
Proposition C.4. \( t \rightarrow^{*}_{pq} \phi(t) \) for all \( t \in \mathcal{T}J^m \).

Proof. This result is proved together with the fact
\[
\phi(t)(\phi(u), l, (x)\phi(v)) \rightarrow^{*}_{pq} s(\phi(t(u, l)), x, \phi(v)),
\]
by simultaneous induction on the structure of \( t \) and \( l \). We show the cases relative to \( l \), where direct use of permutations is required.

Case \( l = [] \).
\[
\phi(t)(\phi(u), [], (x)\phi(v))
= \phi(t)(\phi(u), (x)\phi(v))
\rightarrow^{*}_{pq} \phi(t)[\phi(u), (x)\phi(v)] \quad (\text{Lemma 2.8.2})
\rightarrow^{*}_{pq} s(\phi(t)(\phi(u)), x, \phi(v)) \quad (\text{Lemma C.3})
= s(\phi(t(u, [])), x, \phi(v)) \quad (\text{Lemma C.1.1})
\]

Case \( l = u_1::l_1 \).
\[
\phi(t)(\phi(u), u_1::l_1, (x)\phi(v))
\rightarrow^{u}_{q} \phi(t)(\phi(u))(u_1, l_1, (x)\phi(v))
\rightarrow^{*}_{pq} \phi(t(u)(\phi(u_1), l_1, (x)\phi(v)) \quad (\text{Lemma C.1.1 and IH})
\rightarrow^{*}_{pq} s(\phi(t(u)(u_1, l_1)), x, \phi(v)) \quad (\text{IIH})
= s(\phi(t(u, u_1::l_1)), x, \phi(v)) \quad (\text{Lemma C.1.1})
\]

□

Now we prove the main theorems about permutations. This is done with the help of Propositions C.2 and C.4. We start with the relationship between \( \rightarrow_{pq} \) and the kernel of \( \phi \).

Theorem C.5 Permutability. \( \phi(t_1) = \phi(t_2) \iff t_1 \leftrightarrow^{*}_{pq} t_2 \), for all \( t_1, t_2 \in \mathcal{T}J^m \).

Proof. Proposition C.2 guarantees \( \phi(t_1) = \phi(t_2) \) whenever \( t_1 \leftrightarrow^{*}_{pq} t_2 \). As to the only if part, we use Proposition C.4, obtaining \( t_1 \rightarrow^{*}_{pq} \phi(t_1) \) and \( t_2 \rightarrow^{*}_{pq} \phi(t_2) \) and thus, as by hypothesis \( \phi(t_1) = \phi(t_2) \), \( t_1 \) and \( t_2 \) are inter-permutable. □

The \( pq \)-normal forms are the \( \lambda \)-terms.

Theorem C.6 Characterisation of \( pq \)-Normal Forms. For all \( t \in \mathcal{T}J^m \), \( t \) is \( pq \)-normal iff \( t \in \mathcal{T} \).

Proof. On the one hand, \( \lambda \)-terms have neither \( p \) or \( q \)-redexes and so they are \( pq \)-normal. Consider, on the other hand, that \( t \) is \( pq \)-normal. By Proposition C.4, \( t \rightarrow^{*}_{pq} \phi(t) \) and thus the normality of \( t \) implies \( t = \phi(t) \). The proof concludes observing that the co-domain of \( \phi \) is \( \mathcal{T} \). □

Now the two main properties of relation \( \rightarrow_{pq} \) are established.

Theorem C.7 Confluence. \( \rightarrow_{pq} \) is confluent.

Proof. Assuming \( t \overset{*}_{pq} t_1 \) and \( t \overset{*}_{pq} t_2 \), by Proposition C.4 follows that \( t_1 \overset{*}_{pq} \phi(t_1) \) and \( t_2 \overset{*}_{pq} \phi(t_2) \). Yet by Proposition C.4 we have \( t \overset{*}_{pq} \phi(t) \) and we can now use Proposition C.2 to conclude that \( \phi(t_1) = \phi(t) = \phi(t_2) \). □

Theorem C.8 Termination. \( \rightarrow_{pq} \) is terminating.

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.
have a RHS of weight lower than its LHS and thus every sequence of permutations must be finite.

Note that \( q_J \) respectively of Propositions C.2 and C.4.

By Proposition C.4, \( \lambda J^m \)-term \( t \) is con
tuent and terminating, each permutation can be shown to have a RHS of weight lower than its LHS and thus every sequence of permutations must be finite.

Case \( p_1 \): \( w(t(u,l,(x)v)) = w(t) + w(u) + w(l) + 1 > 1 = w(y) \).

Case \( p_2 \):

\[
\begin{align*}
& w(t(u,l,(x)\lambda y.v)) = (1 + w(v))(w(t) + w(u) + w(l) + 1) \\
& > 1 + w(v)(w(t) + w(u) + w(l) + 1) \\
& \geq w(\lambda y.t[u,l,(x)v]) .
\end{align*}
\]

where the inequality step follows from the fact

\[
\begin{align*}
& w(t_1[u_1,l_1,(x)\lambda y.v]) < (w(t_2) + 1)(w(t_1) + w(u_1) + w(l_1)) + w(l_2),
\end{align*}
\]

which is proved by induction on \( l_2 \).

Case \( q \):

\[
\begin{align*}
& w(t(u,v::l,(x)v')) = w(v')(w(t) + w(u) + 2 + w(v) + w(l) + 1) \\
& > w(v')(w(t) + w(u) + 1 + w(v) + w(l) + 1) \\
& = w(t(u,v,l,(x)v')).
\end{align*}
\]

Since \( \rightarrow_{pq} \) is confluent and terminating, each \( \lambda J^m \)-term \( t \) has a unique normal form that we denote by \( \downarrow_{pq} t \). The rewriting system \( \rightarrow_{pq} \) calculates \( \phi \).

Theorem C.9 Representation of \( \phi \). \( \phi(t) = \downarrow_{pq} t \), for all \( t \in T J^m \).

Proof. By Proposition C.4, \( t \rightarrow_{pq} \phi(t) \) and \( \phi(t) \) is a normal form.

C.2 Other results

Propositions C.10, C.11, C.12 and C.13 below are the analogues to \( p \), \( q \), \( p^m \) and \( qJ \) respectively of Propositions C.2 and C.4.

Proposition C.10. For all \( t,u \in T J \):

1. If \( t \rightarrow_{p}^* u \) then \( p(t) = p(u) \);
2. \( t \rightarrow_{p}^* p(t) \).

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.
PROOF. From Proposition C.2, if \( t \rightarrow^*_p u \), \( \phi(t) = \phi(u) \). The proof of 1 concludes observing that \( p \) is the restriction of \( \phi \) to \( \lambda J \)-terms.

In order to prove 2, observe first that \( t \rightarrow^*_{pq} \phi(t) = p(t) \) by Proposition C.4. The proof concludes observing that \( \lambda J \)-terms have no \( q \)-redexes and \( \lambda J \) is closed for \( p \)-permutations \( \square \)

**Proposition C.11.** For all \( t, u \in T^m \):

1. if \( t \rightarrow^*_2 u \) then \( q(t) = q(u) \);
2. \( t \rightarrow^*_2 q(t) \).

**Proof.** Analogous to the proof of the proposition above. \( \square \)

**Proposition C.12.** For all \( t, u \in T^{1,m} \):

1. if \( t \rightarrow^*_1 u \) then \( p^m(t) = p^m(u) \);
2. \( t \rightarrow^*_1 p^m(t) \).

**Proof.** As to 1, the cases \( p_1 \) and \( p_2 \) follow as the corresponding cases in the proof of Proposition C.2, simply by replacing \( \phi \) by \( p^m \) and with the help of the following analogue of Lemma C.1.2 for \( p^m \),

\[
p^m(t(t(u,l),(x)v)) = s(p^m(t(u,l)),x,p^m(v)) = p^m(t\{u,l,(x)v\}) , \tag{34}
\]

which results by simple calculations. The case \( p_3 \) uses additionally the fact

\[
p'^m(t\{u,l,(x)l_0\}) = s'(p^m(t^m(t(u,l)),p^m(u,l)),x,p^m(l_0)) , \tag{35}
\]

proved by induction on \( l_0 \). The case \( p_3 \) is as follows:

\[
p^m(t_1(u_1,l_1,(x)t_2(u_2,l_2,(y)v))) = s(p^m(t_1(p^m(u_1),p^m(l_1)),x,p^m(t_2(p^m(u_2),p^m(l_2),y)p^m(v))))
\]

\[
= s(p^m(t_1(u_1,l_1,(x)t_2)(p^m(t_1(u_1,l_1,(x)u_2)),p^m(t_1\{u_1,l_1,(x)l_2\},y,p^m(v))))
\]

( Substitution lemma, \( x \notin v \) and fact (35))

\[
= p^m(t_1\{u_1,l_1,(x)l_2\},t_1\{u_1,l_1,(x)u_2\},v_1) (\text{Fact (34)})
\]

Statement 2 of this proposition is proved together with property

\[
l \rightarrow^*_p p'^m(l),
\]

by simultaneous induction on \( t \) and \( l \). We illustrate the case where \( t \) is a \( \text{gm} \)-application.

\[
t_1(u_1,l_1,(x)v_1) \rightarrow^*_p p^m(t_1)(p^m(u_1),p^m(l_1),(x)p^m(v_1)) \tag{IH}
\]

\[
\rightarrow^*_p p^m(t_1\{p^m(u_1),p^m(l_1),(x)p^m(v_1)\}) \tag{Lemma 2.8.2}
\]

\[
\rightarrow^*_p s(p^m(t_1\{p^m(u_1),p^m(l_1),(x)p^m(v_1)\}),x,p^m(v_1)) \tag{Lemma C.3}
\]

\[
= p^m(t_1\{u_1,l_1,(x)v_1\}).
\]

\( \square \)

**Proposition C.13.** For all \( t, u \in T^{1J,m} \):

1. if \( t \rightarrow^*_2 u \) then \( qJ(t) = qJ(u) \);
2. \( t \rightarrow^*_2 q(t) \).

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.
Proof. The proof of 1 is analogous to the case corresponding to permutation $(q)$ in the proof of Proposition C.2. It requires the fact

$$qJ(t(u)) = qJ(t)(qJ(u)),$$

for all $t, u \in TJ^m$.

Statement 2 is proved together with property

$$t(u, l, (x)v) \rightarrow qJ'(t, u, l, x, v),$$

for all $t, u, v \in TJ$ and for all $l \in LJ$

by simultaneous induction on $t$ and $l$; the proof is analogous to the proof of Proposition C.4, but simpler.

The sequence of theorems in Subsection 4.1 (Theorems C.6 to C.9) can now be analogously established for mappings $p, q, p^m$ and $qJ$, with the help of Propositions C.10, C.11, C.12 and C.13.

Theorem C.14. For each of the following combinations of $F, P, S$ and $S'$

<table>
<thead>
<tr>
<th></th>
<th>$F$</th>
<th>$P$</th>
<th>$S$</th>
<th>$S'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p^m$</td>
<td>$p$</td>
<td>$\lambda J^m$</td>
<td>$\lambda m$</td>
<td></td>
</tr>
<tr>
<td>$qJ$</td>
<td>$q$</td>
<td>$\lambda J^m$</td>
<td>$\lambda J$</td>
<td></td>
</tr>
<tr>
<td>$p$</td>
<td>$p$</td>
<td>$\lambda J$</td>
<td>$\lambda$</td>
<td></td>
</tr>
<tr>
<td>$q$</td>
<td>$q$</td>
<td>$\lambda m$</td>
<td>$\lambda$</td>
<td></td>
</tr>
</tbody>
</table>

the properties below hold.

1. For all $t_1, t_2 \in S$, $F(t_1) = F(t_2)$ iff $t_1 \leftrightarrow p t_2$.
2. For all $t \in S$, $t$ is a $P$-normal form iff $t \in S'$.
3. $\rightarrow_p$ is confluent and terminating in $S$ and, for all $t \in S$, $\downarrow_p t = F(t)$. 

ACM Transactions on Computational Logic, Vol. V, No. N, Month 20YY.