

Broadcast Psi-calculi

Palle Raabjerg Johannes Åman Pohjola Tjark Weber

June 17, 2024

Abstract

We provide an Isabelle/HOL-Nominal formalisation of the definitions, theorems and proofs in the paper *Broadcast Psi-calculi with an Application to Wireless Protocols* by Borgström et al., which extends the Psi-calculi framework with primitives for broadcast communication in order to model wireless protocols.

1 Introduction

We provide an Isabelle/HOL-Nominal formalisation of the definitions, theorems and proofs in the paper *Broadcast Psi-calculi with an Application to Wireless Protocols* [4, 5], which extends the Psi-calculi framework [2, 3, 1] with primitives for broadcast communication in order to model wireless protocols.

The file `Broadcast_Thms.thy` contains a collection of the relevant definitions and theorems, with comments relating them directly to the paper.

2 Formalisation

```
theory Broadcast-Chain
imports Psi-Calculi.Chain
begin

lemma pair-perm-fresh-contr:
  fixes a::'a and b::'a
  assumes
    at: at TYPE('a)
  and
    prems: b # pi (a, b) ∈ set pi
  shows False
  ⟨proof⟩

lemma pair-perm-fresh-contr':
  fixes a::'a and b::'a
  assumes
```

```

    at: at TYPE('a)
  and
    prems: a # pi (a, b) ∈ set pi
  shows False
  ⟨proof⟩

lemma list-set-supp:
  fixes l :: ('d::fs-name) list
  shows supp (set l) = (supp l :: name set)
  ⟨proof⟩

lemma name-set-supp:
  assumes finite a
  shows supp a = (a::name set)
  ⟨proof⟩

lemma supp-idem:
  fixes l :: ('d::fs-name)
  shows supp((supp l)::name set) = (supp(l)::name set)
  ⟨proof⟩

lemma fresh-supp:
  fixes a :: name
  and X :: ('d::fs-name)
  shows a # ((supp X)::name set) = a # X
  ⟨proof⟩

lemma fresh-chain-supp:
  fixes A :: name list
  and X :: ('d::fs-name)
  shows A #* ((supp X)::name set) = A #* X
  ⟨proof⟩

lemma fresh-chain-fin-union:
  fixes X::('d::fs-name set)
  and Y::('d::fs-name set)
  and A::name list
  assumes f1: finite X
  and f2: finite Y
  shows A#*(X∪Y) = (A#*X ∧ A#*Y)
  ⟨proof⟩

lemma fresh-subset:
  fixes S :: name set
  and S' :: name set
  and a :: name
  assumes a # S
  and S' ⊆ S
  and finite S

```

shows $a \# S'$
<proof>

lemma *fresh-subset'*:
 fixes $S :: 'd::fs\text{-name set}$
 and $S' :: 'd::fs\text{-name set}$
 and $a :: \text{name}$
 assumes $a \# S$
 and $S' \subseteq S$
 and *finite* S

shows $a \# S'$
<proof>

lemma *fresh-star-subset'*:
 fixes $S :: 'd::fs\text{-name set}$
 and $S' :: 'd::fs\text{-name set}$
 and $A :: \text{name list}$
 assumes $A \#* S$
 and $S' \subseteq S$
 and *finite* S

shows $A \#* S'$
<proof>

lemma *fresh-star-subset*:
 fixes $S :: \text{name set}$
 and $S' :: \text{name set}$
 and $A :: \text{name list}$
 assumes $A \#* S$
 and $S' \subseteq S$
 and *finite* S

shows $A \#* S'$
<proof>

lemma *times-set-fresh*:
 fixes $a :: \text{name}$
 and $S :: \text{name list}$
 and $S' :: \text{name list}$
 assumes $a \# \text{set } S$
 and $a \# \text{set } S'$
 shows $a \# \text{set } S \times \text{set } S'$
<proof>

lemma *times-set-fresh-star*:
 fixes $A :: \text{name list}$
 and $S :: \text{name list}$

```

    and S' :: name list
  assumes A #* set S
    and A #* set S'
  shows A #* (set S × set S')
  ⟨proof⟩

lemma supp-list-set:
  fixes M::'d::fs-name list
  shows (supp M) = ((supp(set M))::name set)
  ⟨proof⟩

lemma fresh-list-set:
  fixes M::'d::fs-name list
    and A::name list
  shows A #* set M = A #* M
  ⟨proof⟩

lemma permSupp:
  fixes Ψ :: name prm
    and Ψ' :: 'd::fs-name

  shows (supp(Ψ · Ψ')::name set) ⊆ ((supp Ψ) ∪ (supp Ψ'))
  ⟨proof⟩

end
theory Broadcast-Frame
  imports Psi-Calculi.Frame
begin

locale assertionAux = Frame.assertionAux SCompose SImp SBottom SChanEq
  for SCompose :: 'b::fs-name ⇒ 'b ⇒ 'b    (infixr ⊗ 80)
  and SImp :: 'b ⇒ 'c::fs-name ⇒ bool      (- ⊢ - [70, 70] 70)
  and SBottom :: 'b                          (⊥ 90)
  and SChanEq :: ('a::fs-name ⇒ 'a ⇒ 'c)    (- ↔ - [80, 80] 80)
  +
  fixes SOutCon :: 'a::fs-name ⇒ 'a ⇒ 'c    (- ≲ - [80, 80] 80)
  and SInCon  :: 'a::fs-name ⇒ 'a ⇒ 'c    (- ≳ - [80, 80] 80)

assumes statEqvt'''[eqvt]: ∧p::name prm. p · (M ≲ N) = (p · M) ≲ (p · N)
  and statEqvt'''[eqvt]: ∧p::name prm. p · (M ≳ N) = (p · M) ≳ (p · N)

begin

lemma chanInConSupp:
  fixes M :: 'a
    and N :: 'a

  shows (supp(M ≳ N)::name set) ⊆ ((supp M) ∪ (supp N))
  ⟨proof⟩

```

```

lemma chanOutConSupp:
  fixes  $M :: 'a$ 
  and  $N :: 'a$ 

shows  $(\text{supp}(M \preceq N) :: \text{name set}) \subseteq ((\text{supp } M) \cup (\text{supp } N))$ 
   $\langle \text{proof} \rangle$ 

lemma freshInCon[intro]:
  fixes  $x :: \text{name}$ 
  and  $M :: 'a$ 
  and  $N :: 'a$ 

assumes  $x \# M$ 
  and  $x \# N$ 

shows  $x \# M \succeq N$ 
   $\langle \text{proof} \rangle$ 

lemma freshInConChain[intro]:
  fixes  $\text{vec} :: \text{name list}$ 
  and  $Xs :: \text{name set}$ 
  and  $M :: 'a$ 
  and  $N :: 'a$ 

shows  $[\![\text{vec} \#^* M; \text{vec} \#^* N]\!] \implies \text{vec} \#^* (M \succeq N)$ 
  and  $[\![Xs \#^* M; Xs \#^* N]\!] \implies Xs \#^* (M \succeq N)$ 
   $\langle \text{proof} \rangle$ 

lemma freshOutCon[intro]:
  fixes  $x :: \text{name}$ 
  and  $M :: 'a$ 
  and  $N :: 'a$ 

assumes  $x \# M$ 
  and  $x \# N$ 

shows  $x \# M \preceq N$ 
   $\langle \text{proof} \rangle$ 

lemma freshOutConChain[intro]:
  fixes  $\text{vec} :: \text{name list}$ 
  and  $Xs :: \text{name set}$ 
  and  $M :: 'a$ 
  and  $N :: 'a$ 

shows  $[\![\text{vec} \#^* M; \text{vec} \#^* N]\!] \implies \text{vec} \#^* (M \preceq N)$ 
  and  $[\![Xs \#^* M; Xs \#^* N]\!] \implies Xs \#^* (M \preceq N)$ 
   $\langle \text{proof} \rangle$ 

```

```

lemma chanOutConClosed:
  fixes  $\Psi :: 'b$ 
    and  $M :: 'a$ 
    and  $N :: 'a$ 
    and  $p :: \text{name prm}$ 

assumes  $\Psi \vdash M \preceq N$ 

shows  $(p \cdot \Psi) \vdash (p \cdot M) \preceq (p \cdot N)$ 
  <proof>

lemma chanInConClosed:
  fixes  $\Psi :: 'b$ 
    and  $M :: 'a$ 
    and  $N :: 'a$ 
    and  $p :: \text{name prm}$ 

assumes  $\Psi \vdash M \succeq N$ 

shows  $(p \cdot \Psi) \vdash (p \cdot M) \succeq (p \cdot N)$ 
  <proof>

end

locale assertion = assertionAux SCompose SImp SBottom SChanEq SOutCon SInCon
  + assertion SCompose SImp SBottom SChanEq
  for SCompose ::  $'b::\text{fs-name} \Rightarrow 'b \Rightarrow 'b$ 
  and SImp ::  $'b \Rightarrow 'c::\text{fs-name} \Rightarrow \text{bool}$ 
  and SBottom ::  $'b$ 
  and SChanEq ::  $'a::\text{fs-name} \Rightarrow 'a \Rightarrow 'c$ 
  and SOutCon ::  $'a::\text{fs-name} \Rightarrow 'a \Rightarrow 'c$ 
  and SInCon ::  $'a::\text{fs-name} \Rightarrow 'a \Rightarrow 'c$  +

  assumes chanOutConSupp: SImp  $\Psi$  (SOutCon  $M$   $N$ )  $\Longrightarrow$  (((supp  $N$ )::name set)
   $\subseteq$  ((supp  $M$ )::name set))
  and chanInConSupp: SImp  $\Psi$  (SInCon  $N$   $M$ )  $\Longrightarrow$  (((supp  $N$ )::name set)  $\subseteq$ 
  ((supp  $M$ )::name set))

begin

notation SOutCon ( $- \preceq -$  [90, 90] 90)
notation SInCon ( $- \succeq -$  [90, 90] 90)

end

end

theory Semantics
  imports Broadcast-Chain Broadcast-Frame

```

begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Semantics* from [1]. The nominal datatypes (*'a, 'b, 'c*) *residual* and *'a action* have been extended with constructors for broadcast input and output. This leads to a different semantics.

nominal-datatype (*'a, 'b, 'c*) *boundOutput* =
BOut 'a::fs-name ('a, 'b::fs-name, 'c::fs-name) psi (- <' - [110, 110] 110)
| *BStep «name» ('a, 'b, 'c) boundOutput ((ν-)- [110, 110] 110)*

primrec *BOresChain :: name list ⇒ ('a::fs-name, 'b::fs-name, 'c::fs-name) bound-Output ⇒*
('a, 'b, 'c) boundOutput

where

Base: BOresChain [] B = B
| *Step: BOresChain (x#xs) B = (νx)(BOresChain xs B)*

abbreviation

BOresChainJudge ((ν-)- [80, 80] 80) where ((ν*xvec)B ≡ BOresChain xvec B*

lemma *BOresChainEqvt[eqvt]:*

fixes *perm :: name prm*
and *lst :: name list*
and *B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput*

shows *perm · ((ν*xvec)B) = (ν*(perm · xvec))(perm · B)*
<proof>

lemma *BOresChainSimps[simp]:*

fixes *xvec :: name list*
and *N :: 'a::fs-name*
and *P :: ('a, 'b::fs-name, 'c::fs-name) psi*
and *N' :: 'a*
and *P' :: ('a, 'b, 'c) psi*
and *B :: ('a, 'b, 'c) boundOutput*
and *B' :: ('a, 'b, 'c) boundOutput*

shows *((ν*xvec)N <' P = N' <' P') = (xvec = [] ∧ N = N' ∧ P = P')*
and *(N' <' P' = (ν*xvec)N <' P) = (xvec = [] ∧ N = N' ∧ P = P')*
and *(N' <' P' = N <' P) = (N = N' ∧ P = P')*
and *((ν*xvec)B = (ν*xvec)B') = (B = B')*
<proof>

lemma *outputFresh[simp]:*

fixes *Xs :: name set*
and *xvec :: name list*
and *N :: 'a::fs-name*
and *P :: ('a, 'b::fs-name, 'c::fs-name) psi*

shows $(Xs \#* (N \prec' P)) = ((Xs \#* N) \wedge (Xs \#* P))$
and $(xvec \#* (N \prec' P)) = ((xvec \#* N) \wedge (xvec \#* P))$
 $\langle proof \rangle$

lemma *boundOutputFresh*:
fixes $x :: name$
and $xvec :: name list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$

shows $(x \#* ((\nu * xvec) B)) = (x \in set xvec \vee x \# B)$
 $\langle proof \rangle$

lemma *boundOutputFreshSet*:
fixes $Xs :: name set$
and $xvec :: name list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$
and $yvec :: name list$
and $x :: name$

shows $Xs \#* ((\nu * xvec) B) = (\forall x \in Xs. x \in set xvec \vee x \# B)$
and $yvec \#* ((\nu * xvec) B) = (\forall x \in (set yvec). x \in set xvec \vee x \# B)$
and $Xs \#* ((\nu x) B) = Xs \#* [x].B$
and $xvec \#* ((\nu x) B) = xvec \#* [x].B$
 $\langle proof \rangle$

lemma *BOresChainSupp*:
fixes $xvec :: name list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$

shows $(supp((\nu * xvec) B)::name set) = (supp B) - (supp xvec)$
 $\langle proof \rangle$

lemma *boundOutputFreshSimps[simp]*:
fixes $Xs :: name set$
and $xvec :: name list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$
and $yvec :: name list$
and $x :: name$

shows $Xs \#* xvec \implies (Xs \#* ((\nu * xvec) B)) = (Xs \#* B)$
and $yvec \#* xvec \implies yvec \#* ((\nu * xvec) B) = yvec \#* B$
and $xvec \#* ((\nu * xvec) B)$
and $x \# xvec \implies x \# ((\nu * xvec) B) = x \# B$
 $\langle proof \rangle$

lemma *boundOutputChainAlpha*:
fixes $p :: name prm$
and $xvec :: name list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$

and $yvec :: name\ list$

assumes $xvecFreshB: (p \cdot xvec) \#* B$
and $S: set\ p \subseteq set\ xvec \times set\ (p \cdot xvec)$
and $(set\ xvec) \subseteq (set\ yvec)$

shows $((\nu* yvec) B) = ((\nu*(p \cdot yvec))(p \cdot B))$
 $\langle proof \rangle$

lemma *boundOutputChainAlpha'*:
fixes $p :: name\ prm$
and $xvec :: name\ list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name)\ boundOutput$
and $yvec :: name\ list$
and $zvec :: name\ list$

assumes $xvecFreshB: xvec \#* B$
and $S: set\ p \subseteq set\ xvec \times set\ yvec$
and $yvec \#* ((\nu*zvec) B)$

shows $((\nu*zvec) B) = ((\nu*(p \cdot zvec))(p \cdot B))$
 $\langle proof \rangle$

lemma *boundOutputChainAlpha''*:
fixes $p :: name\ prm$
and $xvec :: name\ list$
and $M :: 'a::fs-name$
and $P :: ('a::fs-name, 'b::fs-name, 'c::fs-name)\ psi$
and $yvec :: name\ list$

assumes $(p \cdot xvec) \#* M$
and $(p \cdot xvec) \#* P$
and $set\ p \subseteq set\ xvec \times set\ (p \cdot xvec)$
and $(set\ xvec) \subseteq (set\ yvec)$

shows $((\nu* yvec) M \prec' P) = ((\nu*(p \cdot yvec))(p \cdot M) \prec' (p \cdot P))$
 $\langle proof \rangle$

lemma *boundOutputChainSwap*:
fixes $x :: name$
and $y :: name$
and $N :: 'a::fs-name$
and $P :: ('a, 'b::fs-name, 'c::fs-name)\ psi$
and $xvec :: name\ list$

assumes $y \# N$
and $y \# P$
and $x \in (set\ xvec)$

shows $(\nu * xvec)N \prec' P = (\nu * ([x, y] \cdot xvec))([x, y] \cdot N) \prec' ([x, y] \cdot P)$
 $\langle proof \rangle$

lemma *alphaBoundOutput*:

fixes $x :: name$
and $y :: name$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$

assumes $y \# B$

shows $(\nu x)B = (\nu y)([x, y] \cdot B)$
 $\langle proof \rangle$

lemma *boundOutputEqFresh*:

fixes $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$
and $C :: ('a, 'b, 'c) boundOutput$
and $x :: name$
and $y :: name$

assumes $(\nu x)B = (\nu y)C$
and $x \# B$

shows $y \# C$
 $\langle proof \rangle$

lemma *boundOutputEqSupp*:

fixes $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$
and $C :: ('a, 'b, 'c) boundOutput$
and $x :: name$
and $y :: name$

assumes $(\nu x)B = (\nu y)C$
and $x \in supp B$

shows $y \in supp C$
 $\langle proof \rangle$

lemma *boundOutputChainEq*:

fixes $xvec :: name list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput$
and $yvec :: name list$
and $B' :: ('a, 'b, 'c) boundOutput$

assumes $(\nu * xvec)B = (\nu * yvec)B'$
and $xvec \#* yvec$
and $length xvec = length yvec$

shows $\exists p. (set p) \subseteq (set xvec) \times set (yvec) \wedge distinctPerm p \wedge B = p \cdot B' \wedge$
 $(set (map fst p)) \subseteq (supp B) \wedge xvec \#* B' \wedge yvec \#* B$

<proof>

lemma *boundOutputChainEqLength*:

fixes *xvec* :: *name list*
and *M* :: '*a*::*fs-name*
and *P* :: ('*a*, '*b*::*fs-name*, '*c*::*fs-name*) *psi*
and *yvec* :: *name list*
and *N* :: '*a*::*fs-name*
and *Q* :: ('*a*, '*b*::*fs-name*, '*c*::*fs-name*) *psi*

assumes $(\nu*xvec)M \prec' P = (\nu*yvec)N \prec' Q$

shows $length\ xvec = length\ yvec$

<proof>

lemma *boundOutputChainEq'*:

fixes *xvec* :: *name list*
and *M* :: '*a*::*fs-name*
and *P* :: ('*a*, '*b*::*fs-name*, '*c*::*fs-name*) *psi*
and *yvec* :: *name list*
and *N* :: '*a*
and *Q* :: ('*a*::*fs-name*, '*b*::*fs-name*, '*c*::*fs-name*) *psi*

assumes $(\nu*xvec)M \prec' P = (\nu*yvec)N \prec' Q$

and $xvec \#* yvec$

shows $\exists p. (set\ p) \subseteq (set\ xvec) \times set\ (yvec) \wedge distinctPerm\ p \wedge M = p \cdot N \wedge P = p \cdot Q \wedge xvec \#* N \wedge xvec \#* Q \wedge yvec \#* M \wedge yvec \#* P$

<proof>

lemma *boundOutputChainEq''*:

fixes *xvec* :: *name list*
and *M* :: '*a*::*fs-name*
and *P* :: ('*a*, '*b*::*fs-name*, '*c*::*fs-name*) *psi*
and *yvec* :: *name list*
and *N* :: '*a*
and *Q* :: ('*a*::*fs-name*, '*b*::*fs-name*, '*c*::*fs-name*) *psi*

assumes $(\nu*xvec)M \prec' P = (\nu*yvec)N \prec' Q$

and $xvec \#* yvec$

and *distinct* *xvec*

and *distinct* *yvec*

obtains *p* **where** $(set\ p) \subseteq (set\ xvec) \times set\ (p \cdot xvec)$ **and** *distinctPerm* *p* **and** $yvec = p \cdot xvec$ **and** $N = p \cdot M$ **and** $Q = p \cdot P$ **and** $xvec \#* N$ **and** $xvec \#* Q$ **and** $(p \cdot xvec) \#* M$ **and** $(p \cdot xvec) \#* P$

<proof>

lemma *boundOutputEqSupp'*:

```

fixes  $x$   :: name
and  $xvec$  :: name list
and  $M$     :: 'a::fs-name
and  $P$     :: ('a, 'b::fs-name, 'c::fs-name) psi
and  $y$     :: name
and  $yvec$  :: name list
and  $N$     :: 'a
and  $Q$     :: ('a, 'b, 'c) psi

assumes  $Eq: (\nu x)(\nu *xvec)M \prec' P = (\nu y)(\nu *yvec)N \prec' Q$ 
and  $x \neq y$ 
and  $x \# yvec$ 
and  $x \# xvec$ 
and  $y \# xvec$ 
and  $y \# yvec$ 
and  $xvec \#* yvec$ 
and  $x \in \text{supp } M$ 

shows  $y \in \text{supp } N$ 
 $\langle \text{proof} \rangle$ 

lemma boundOutputChainOpenIH:
fixes  $xvec$  :: name list
and  $x$      :: name
and  $B$      :: ('a::fs-name, 'b::fs-name, 'c::fs-name) boundOutput
and  $yvec$   :: name list
and  $y$      :: name
and  $B'$     :: ('a, 'b, 'c) boundOutput

assumes  $Eq: (\nu *xvec)(\nu x)B = (\nu *yvec)(\nu y)B'$ 
and  $L: \text{length } xvec = \text{length } yvec$ 
and  $xFreshB': x \# B'$ 
and  $xFreshxvec: x \# xvec$ 
and  $xFreshyvec: x \# yvec$ 

shows  $(\nu *xvec)B = (\nu *yvec)((x, y) \cdot B')$ 
 $\langle \text{proof} \rangle$ 

lemma boundOutputPar1Dest:
fixes  $xvec$  :: name list
and  $M$      :: 'a::fs-name
and  $P$      :: ('a, 'b::fs-name, 'c::fs-name) psi
and  $yvec$   :: name list
and  $N$      :: 'a
and  $Q$      :: ('a, 'b, 'c) psi
and  $R$      :: ('a, 'b, 'c) psi

assumes  $(\nu *xvec)M \prec' P = (\nu *yvec)N \prec' (Q \parallel R)$ 
and  $xvec \#* R$ 

```

and $yvec \#* R$

obtains T **where** $P = T \parallel R$ **and** $(\nu*xvec)M \prec' T = (\nu*yvec)N \prec' Q$
 $\langle proof \rangle$

lemma *boundOutputPar1Dest'*:

fixes $xvec :: name\ list$
and $M :: 'a::fs-name$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$
and $yvec :: name\ list$
and $N :: 'a$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

assumes $(\nu*xvec)M \prec' P = (\nu*yvec)N \prec' (Q \parallel R)$
and $xvec \#* yvec$

obtains $T\ p$ **where** $set\ p \subseteq set\ xvec \times set\ yvec$ **and** $P = T \parallel (p \cdot R)$ **and**
 $(\nu*xvec)M \prec' T = (\nu*yvec)N \prec' Q$
 $\langle proof \rangle$

lemma *boundOutputPar2Dest*:

fixes $xvec :: name\ list$
and $M :: 'a::fs-name$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$
and $yvec :: name\ list$
and $N :: 'a$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

assumes $(\nu*xvec)M \prec' P = (\nu*yvec)N \prec' (Q \parallel R)$
and $xvec \#* Q$
and $yvec \#* Q$

obtains T **where** $P = Q \parallel T$ **and** $(\nu*xvec)M \prec' T = (\nu*yvec)N \prec' R$
 $\langle proof \rangle$

lemma *boundOutputPar2Dest'*:

fixes $xvec :: name\ list$
and $M :: 'a::fs-name$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$
and $yvec :: name\ list$
and $N :: 'a$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

assumes $(\nu*xvec)M \prec' P = (\nu*yvec)N \prec' (Q \parallel R)$
and $xvec \#* yvec$

obtains T p **where** $set\ p \subseteq set\ xvec \times set\ yvec$ **and** $P = (p \cdot Q) \parallel T$ **and**
 $(\nu*xvec)M \prec' T = (\nu*yvec)N \prec' R$
 $\langle proof \rangle$

lemma *boundOutputApp*:
fixes $xvec :: name\ list$
and $yvec :: name\ list$
and $B :: ('a::fs-name, 'b::fs-name, 'c::fs-name)\ boundOutput$

shows $(\nu*(xvec@yvec))B = (\nu*xvec)((\nu*yvec)B)$
 $\langle proof \rangle$

lemma *openInjectAux*:
fixes $xvec1 :: name\ list$
and $x :: name$
and $xvec2 :: name\ list$
and $yvec :: name\ list$

assumes $length(xvec1@x\#xvec2) = length\ yvec$

shows $\exists yvec1\ y\ yvec2. yvec = yvec1@y\#yvec2 \wedge length\ xvec1 = length\ yvec1 \wedge$
 $length\ xvec2 = length\ yvec2$
 $\langle proof \rangle$

lemma *boundOutputOpenDest*:
fixes $yvec :: name\ list$
and $M :: 'a::fs-name$
and $P :: ('a, 'b::fs-name, 'c::fs-name)\ psi$
and $xvec1 :: name\ list$
and $x :: name$
and $xvec2 :: name\ list$
and $N :: 'a$
and $Q :: ('a, 'b, 'c)\ psi$

assumes $Eq: (\nu*(xvec1@x\#xvec2))M \prec' P = (\nu*yvec)N \prec' Q$
and $x \# xvec1$
and $x \# yvec$
and $x \# N$
and $x \# Q$
and $distinct\ yvec$

obtains $yvec1\ y\ yvec2$ **where** $yvec=yvec1@y\#yvec2$ **and** $length\ xvec1 = length$
 $yvec1$ **and** $length\ xvec2 = length\ yvec2$
and $(\nu*(xvec1@xvec2))M \prec' P = (\nu*(yvec1@yvec2))([(x, y)] \cdot N) \prec' ([(x, y)]$
 $\cdot Q)$
 $\langle proof \rangle$

lemma *boundOutputOpenDest'*:

```

fixes  $yvec$  :: name list
  and  $M$     :: 'a::fs-name
  and  $P$     :: ('a, 'b::fs-name, 'c::fs-name) psi
  and  $xvec1$  :: name list
  and  $x$      :: name
  and  $xvec2$  :: name list
  and  $N$     :: 'a
  and  $Q$     :: ('a, 'b, 'c) psi

```

```

assumes  $Eq$ :  $(\nu^*(xvec1 @ x \# xvec2)) M \prec' P = (\nu^* yvec) N \prec' Q$ 
  and  $x \# xvec1$ 
  and  $x \# yvec$ 
  and  $x \# N$ 
  and  $x \# Q$ 

```

```

obtains  $yvec1$   $y$   $yvec2$  where  $yvec = yvec1 @ y \# yvec2$  and  $length\ xvec1 = length\ yvec1$ 
and  $length\ xvec2 = length\ yvec2$ 
  and  $(\nu^*(xvec1 @ xvec2)) M \prec' P = (\nu^*(yvec1 @ [(x, y)] \cdot yvec2)) ([ (x, y)] \cdot N) \prec'$ 
 $([(x, y)] \cdot Q)$ 
   $\langle proof \rangle$ 

```

```

lemma  $boundOutputScopeDest$ :
  fixes  $xvec$  :: name list
    and  $M$     :: 'a::fs-name
    and  $P$     :: ('a, 'b::fs-name, 'c::fs-name) psi
    and  $yvec$  :: name list
    and  $N$     :: 'a
    and  $x$     :: name
    and  $Q$     :: ('a, 'b, 'c) psi

```

```

assumes  $(\nu^* xvec) M \prec' P = (\nu^* yvec) N \prec' (\nu z) Q$ 
  and  $z \# xvec$ 
  and  $z \# yvec$ 

```

```

obtains  $R$  where  $P = (\nu z) R$  and  $(\nu^* xvec) M \prec' R = (\nu^* yvec) N \prec' Q$ 
   $\langle proof \rangle$ 

```

```

nominal-datatype ('a, 'b, 'c) residual =
   $RIn$  'a::fs-name 'a ('a, 'b::fs-name, 'c::fs-name) psi
  |  $RBrIn$  'a::fs-name 'a ('a, 'b::fs-name, 'c::fs-name) psi
  |  $ROut$  'a ('a, 'b, 'c) boundOutput
  |  $RBrOut$  'a ('a, 'b, 'c) boundOutput
  |  $RTau$  ('a, 'b, 'c) psi

```

```

nominal-datatype 'a action =  $In$  'a::fs-name 'a      (-(|-) [90, 90] 90)
  |  $BrIn$  'a::fs-name 'a      (j-(|-) [90, 90] 90)
  |  $Out$  'a::fs-name name list 'a  (-(|\nu*-|) (-) [90, 90, 90] 90)
  |  $BrOut$  'a::fs-name name list 'a (j-(|\nu*-|) (-) [90, 90, 90] 90)

```

| *Tau* (τ 90)

nominal-primrec *bn* :: ('a::fs-name) action \Rightarrow name list

where

bn ($M(N)$) = []
| *bn* ($iM(N)$) = []
| *bn* ($M(\nu*xvec)\langle N \rangle$) = *xvec*
| *bn* ($iM(\nu*xvec)\langle N \rangle$) = *xvec*
| *bn* (τ) = []
\langle proof \rangle

lemma *bnEqvt[eqvt]*:

fixes *p* :: name prm

and α :: ('a::fs-name) action

shows ($p \cdot bn \alpha$) = $bn(p \cdot \alpha)$

\langle proof \rangle

nominal-primrec *create-residual* :: ('a::fs-name) action \Rightarrow ('a, 'b::fs-name, 'c::fs-name)
psi \Rightarrow ('a, 'b, 'c) residual (- < - [80, 80] 80)

where

$(M(N)) < P = RIn M N P$
| $(iM(N)) < P = RBrIn M N P$
| $M(\nu*xvec)\langle N \rangle < P = ROut M ((\nu*xvec)\langle N \rangle <' P)$
| $(iM(\nu*xvec)\langle N \rangle) < P = RBrOut M ((\nu*xvec)\langle N \rangle <' P)$
| $\tau < P = (RTau P)$
\langle proof \rangle

nominal-primrec *subject* :: ('a::fs-name) action \Rightarrow 'a option

where

subject ($M(N)$) = *Some M*
| *subject* ($iM(N)$) = *Some M*
| *subject* ($M(\nu*xvec)\langle N \rangle$) = *Some M*
| *subject* ($iM(\nu*xvec)\langle N \rangle$) = *Some M*
| *subject* (τ) = *None*
\langle proof \rangle

nominal-primrec *object* :: ('a::fs-name) action \Rightarrow 'a option

where

object ($M(N)$) = *Some N*
| *object* ($iM(N)$) = *Some N*
| *object* ($M(\nu*xvec)\langle N \rangle$) = *Some N*
| *object* ($iM(\nu*xvec)\langle N \rangle$) = *Some N*
| *object* (τ) = *None*
\langle proof \rangle

lemma *optionFreshChain[simp]*:

fixes *xvec* :: name list

and *X* :: name set

shows $xvec \#* (Some\ x) = xvec \#* x$
and $X \#* (Some\ x) = X \#* x$
and $xvec \#* None$
and $X \#* None$
 $\langle proof \rangle$

lemmas $[simp] = fresh\text{-}some\ fresh\text{-}none$

lemma $actionFresh[simp]$:
fixes $x :: name$
and $\alpha :: ('a::fs\text{-}name)\ action$

shows $(x \# \alpha) = (x \# (subject\ \alpha) \wedge x \# (bn\ \alpha) \wedge x \# (object\ \alpha))$
 $\langle proof \rangle$

lemma $actionFreshChain[simp]$:
fixes $X :: name\ set$
and $\alpha :: ('a::fs\text{-}name)\ action$
and $xvec :: name\ list$

shows $(X \#* \alpha) = (X \#* (subject\ \alpha) \wedge X \#* (bn\ \alpha) \wedge X \#* (object\ \alpha))$
and $(xvec \#* \alpha) = (xvec \#* (subject\ \alpha) \wedge xvec \#* (bn\ \alpha) \wedge xvec \#* (object\ \alpha))$
 $\langle proof \rangle$

lemma $subjectEqvt[eqvt]$:
fixes $p :: name\ prm$
and $\alpha :: ('a::fs\text{-}name)\ action$

shows $(p \cdot subject\ \alpha) = subject(p \cdot \alpha)$
 $\langle proof \rangle$

lemma $okjectEqvt[eqvt]$:
fixes $p :: name\ prm$
and $\alpha :: ('a::fs\text{-}name)\ action$

shows $(p \cdot object\ \alpha) = object(p \cdot \alpha)$
 $\langle proof \rangle$

lemma $create\text{-}residualEqvt[eqvt]$:
fixes $p :: name\ prm$
and $\alpha :: ('a::fs\text{-}name)\ action$
and $P :: ('a, 'b::fs\text{-}name, 'c::fs\text{-}name)\ psi$

shows $(p \cdot (\alpha \prec P)) = (p \cdot \alpha) \prec (p \cdot P)$
 $\langle proof \rangle$

lemma $residualFresh$:
fixes $x :: name$

and $\alpha :: 'a::fs\text{-name}\ action$
and $P :: ('a, 'b::fs\text{-name}, 'c::fs\text{-name})\ psi$

shows $(x \# (\alpha \prec P)) = (x \# (\text{subject } \alpha \wedge (x \in (\text{set}(\text{bn}(\alpha)))) \vee (x \# \text{object}(\alpha) \wedge x \# P))$
 $\langle \text{proof} \rangle$

lemma *residualFresh2[simp]*:
fixes $x :: name$
and $\alpha :: ('a::fs\text{-name})\ action$
and $P :: ('a, 'b::fs\text{-name}, 'c::fs\text{-name})\ psi$

assumes $x \# \alpha$
and $x \# P$

shows $x \# \alpha \prec P$
 $\langle \text{proof} \rangle$

lemma *residualFreshChain2[simp]*:
fixes $xvec :: name\ list$
and $X :: name\ set$
and $\alpha :: ('a::fs\text{-name})\ action$
and $P :: ('a, 'b::fs\text{-name}, 'c::fs\text{-name})\ psi$

shows $\llbracket xvec \#* \alpha; xvec \#* P \rrbracket \Longrightarrow xvec \#* (\alpha \prec P)$
and $\llbracket X \#* \alpha; X \#* P \rrbracket \Longrightarrow X \#* (\alpha \prec P)$
 $\langle \text{proof} \rangle$

lemma *residualFreshSimp[simp]*:
fixes $x :: name$
and $M :: 'a::fs\text{-name}$
and $N :: 'a$
and $P :: ('a, 'b::fs\text{-name}, 'c::fs\text{-name})\ psi$

shows $x \# (M \langle N \rangle \prec P) = (x \# M \wedge x \# N \wedge x \# P)$
and $x \# (j\ M \langle N \rangle \prec P) = (x \# M \wedge x \# N \wedge x \# P)$
and $x \# (M \langle \nu * xvec \rangle \langle N \rangle \prec P) = (x \# M \wedge x \# ((\nu * xvec) \langle N \rangle \prec' P))$
and $x \# (i\ M \langle \nu * xvec \rangle \langle N \rangle \prec P) = (x \# M \wedge x \# ((\nu * xvec) \langle N \rangle \prec' P))$
and $x \# (\tau \prec P) = (x \# P)$
 $\langle \text{proof} \rangle$

lemma *residualInject'*:

shows $(\alpha \prec P = RIn\ M\ N\ Q) = (P = Q \wedge \alpha = M \langle N \rangle)$
and $(\alpha \prec P = RBrIn\ M\ N\ Q) = (P = Q \wedge \alpha = j\ M \langle N \rangle)$
and $(\alpha \prec P = ROut\ M\ B) = (\exists xvec\ N. \alpha = M \langle \nu * xvec \rangle \langle N \rangle \wedge B = (\nu * xvec) \langle N \rangle \prec' P)$
and $(\alpha \prec P = RBrOut\ M\ B) = (\exists xvec\ N. \alpha = i\ M \langle \nu * xvec \rangle \langle N \rangle \wedge B = (\nu * xvec) \langle N \rangle \prec' P)$

$\prec' P)$
and $(\alpha \prec P = RTau Q) = (\alpha = \tau \wedge P = Q)$
and $(RIn M N Q = \alpha \prec P) = (P = Q \wedge \alpha = M(N))$
and $(RBrIn M N Q = \alpha \prec P) = (P = Q \wedge \alpha = \iota M(N))$
and $(ROut M B = \alpha \prec P) = (\exists xvec N. \alpha = M(\nu*xvec)\langle N \rangle \wedge B = (\nu*xvec)\langle N \rangle \prec' P)$
and $(RBrOut M B = \alpha \prec P) = (\exists xvec N. \alpha = \iota M(\nu*xvec)\langle N \rangle \wedge B = (\nu*xvec)\langle N \rangle \prec' P)$
and $(RTau Q = \alpha \prec P) = (\alpha = \tau \wedge P = Q)$
 $\langle proof \rangle$

lemma *residualFreshChainSimp[simp]*:

fixes $xvec :: name\ list$
and $X :: name\ set$
and $M :: 'a::fs-name$
and $N :: 'a$
and $yvec :: name\ list$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

shows $xvec \#* (M(N) \prec P) = (xvec \#* M \wedge xvec \#* N \wedge xvec \#* P)$
and $xvec \#* (\iota M(N) \prec P) = (xvec \#* M \wedge xvec \#* N \wedge xvec \#* P)$
and $xvec \#* (M(\nu*yvec)\langle N \rangle \prec P) = (xvec \#* M \wedge xvec \#* ((\nu*yvec)\langle N \rangle \prec' P))$
and $xvec \#* (\iota M(\nu*yvec)\langle N \rangle \prec P) = (xvec \#* M \wedge xvec \#* ((\nu*yvec)\langle N \rangle \prec' P))$
and $xvec \#* (\tau \prec P) = (xvec \#* P)$
and $X \#* (M(N) \prec P) = (X \#* M \wedge X \#* N \wedge X \#* P)$
and $X \#* (\iota M(N) \prec P) = (X \#* M \wedge X \#* N \wedge X \#* P)$
and $X \#* (M(\nu*yvec)\langle N \rangle \prec P) = (X \#* M \wedge X \#* ((\nu*yvec)\langle N \rangle \prec' P))$
and $X \#* (\iota M(\nu*yvec)\langle N \rangle \prec P) = (X \#* M \wedge X \#* ((\nu*yvec)\langle N \rangle \prec' P))$
and $X \#* (\tau \prec P) = (X \#* P)$
 $\langle proof \rangle$

lemma *residualFreshChainSimp2[simp]*:

fixes $xvec :: name\ list$
and $X :: name\ set$
and $M :: 'a::fs-name$
and $N :: 'a$
and $yvec :: name\ list$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

shows $xvec \#* (RIn M N P) = (xvec \#* M \wedge xvec \#* N \wedge xvec \#* P)$
and $xvec \#* (RBrIn M N P) = (xvec \#* M \wedge xvec \#* N \wedge xvec \#* P)$
and $xvec \#* (ROut M B) = (xvec \#* M \wedge xvec \#* B)$
and $xvec \#* (RBrOut M B) = (xvec \#* M \wedge xvec \#* B)$
and $xvec \#* (RTau P) = (xvec \#* P)$
and $X \#* (RIn M N P) = (X \#* M \wedge X \#* N \wedge X \#* P)$
and $X \#* (RBrIn M N P) = (X \#* M \wedge X \#* N \wedge X \#* P)$
and $X \#* (ROut M B) = (X \#* M \wedge X \#* B)$
and $X \#* (RBrOut M B) = (X \#* M \wedge X \#* B)$
and $X \#* (RTau P) = (X \#* P)$

$\langle proof \rangle$

lemma *freshResidual3*[*dest*]:
fixes $x :: name$
and $\alpha :: ('a::fs-name) action$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

assumes $x \# bn \alpha$
and $x \# \alpha \prec P$

shows $x \# \alpha$ **and** $x \# P$
 $\langle proof \rangle$

lemma *freshResidualChain3*[*dest*]:
fixes $xvec :: name list$
and $\alpha :: ('a::fs-name) action$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

assumes $xvec \#* (\alpha \prec P)$
and $xvec \#* bn \alpha$

shows $xvec \#* \alpha$ **and** $xvec \#* P$
 $\langle proof \rangle$

lemma *freshResidual4*[*dest*]:
fixes $x :: name$
and $\alpha :: ('a::fs-name) action$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

assumes $x \# \alpha \prec P$

shows $x \# subject \alpha$
 $\langle proof \rangle$

lemma *freshResidualChain4*[*dest*]:
fixes $xvec :: name list$
and $\alpha :: ('a::fs-name) action$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

assumes $xvec \#* (\alpha \prec P)$

shows $xvec \#* subject \alpha$
 $\langle proof \rangle$

lemma *alphaOutputResidual*:
fixes $M :: 'a::fs-name$
and $xvec :: name list$
and $N :: 'a$
and $P :: ('a, 'b::fs-name, 'c::fs-name) psi$

and $p \quad :: \text{ name prm}$

assumes $(p \cdot xvec) \#* N$
and $(p \cdot xvec) \#* P$
and $set\ p \subseteq set\ xvec \times set(p \cdot xvec)$
and $set\ xvec \subseteq set\ yvec$

shows $M(\nu^*yvec)\langle N \rangle \prec P = M(\nu^*(p \cdot yvec))\langle (p \cdot N) \rangle \prec (p \cdot P)$
and $iM(\nu^*yvec)\langle N \rangle \prec P = iM(\nu^*(p \cdot yvec))\langle (p \cdot N) \rangle \prec (p \cdot P)$
 $\langle proof \rangle$

lemmas $[simp\ del] = create-residual.simps$

lemma $residualInject''$:

assumes $bn\ \alpha = bn\ \beta$

shows $(\alpha \prec P = \beta \prec Q) = (\alpha = \beta \wedge P = Q)$
 $\langle proof \rangle$

lemmas $residualInject = residual.inject\ create-residual.simps\ residualInject'\ residualInject''$

lemma $bnFreshResidual[simp]$:
fixes $\alpha :: ('a::fs-name)\ action$

shows $(bn\ \alpha) \#* (\alpha \prec P) = bn\ \alpha \#* (subject\ \alpha)$
 $\langle proof \rangle$

lemma $actionCases[case-names\ cInput\ cBrInput\ cOutput\ cBrOutput\ cTau]$:
fixes $\alpha :: ('a::fs-name)\ action$

assumes $\bigwedge M\ N. \alpha = M(\langle N \rangle) \implies Prop$
and $\bigwedge M\ N. \alpha = iM(\langle N \rangle) \implies Prop$
and $\bigwedge M\ xvec\ N. \alpha = M(\nu^*xvec)\langle N \rangle \implies Prop$
and $\bigwedge M\ xvec\ N. \alpha = iM(\nu^*xvec)\langle N \rangle \implies Prop$
and $\alpha = \tau \implies Prop$

shows $Prop$
 $\langle proof \rangle$

lemma $actionPar1Dest$:
fixes $\alpha :: ('a::fs-name)\ action$
and $P :: ('a, 'b::fs-name, 'c::fs-name)\ psi$
and $\beta :: ('a::fs-name)\ action$
and $Q :: ('a, 'b, 'c)\ psi$
and $R :: ('a, 'b, 'c)\ psi$

assumes $\alpha \prec P = \beta \prec (Q \parallel R)$

and $bn\ \alpha \#* bn\ \beta$

obtains $T\ p$ **where** $set\ p \subseteq set(bn\ \alpha) \times set(bn\ \beta)$ **and** $P = T \parallel (p \cdot R)$ **and** $\alpha \prec T = \beta \prec Q$
<proof>

lemma *actionPar2Dest*:
fixes $\alpha :: ('a::fs-name)\ action$
and $P :: ('a, 'b::fs-name, 'c::fs-name)\ psi$
and $\beta :: ('a::fs-name)\ action$
and $Q :: ('a, 'b, 'c)\ psi$
and $R :: ('a, 'b, 'c)\ psi$

assumes $\alpha \prec P = \beta \prec (Q \parallel R)$
and $bn\ \alpha \#* bn\ \beta$

obtains $T\ p$ **where** $set\ p \subseteq set(bn\ \alpha) \times set(bn\ \beta)$ **and** $P = (p \cdot Q) \parallel T$ **and** $\alpha \prec T = \beta \prec R$
<proof>

lemma *actionScopeDest*:
fixes $\alpha :: ('a::fs-name)\ action$
and $P :: ('a, 'b::fs-name, 'c::fs-name)\ psi$
fixes $\beta :: ('a::fs-name)\ action$
and $x :: name$
and $Q :: ('a, 'b, 'c)\ psi$

assumes $\alpha \prec P = \beta \prec (\nu x)Q$
and $x \# bn\ \alpha$
and $x \# bn\ \beta$

obtains R **where** $P = (\nu x)R$ **and** $\alpha \prec R = \beta \prec Q$
<proof>

lemma *emptyFreshName*:
fixes $x :: name$
and $M :: 'a::fs-name$

assumes $supp\ M = (\{\}::name\ set)$

shows $x \# M$
<proof>

lemma *emptyFresh*:
fixes $xvec :: name\ list$
and $M :: 'a::fs-name$

assumes $supp\ M = (\{\}::name\ set)$

```

shows xvec #* M
  ⟨proof⟩

lemma permEmptyEq:
  fixes p :: name prm
    and M :: 'a::fs-name'

assumes suppE: supp M = ({ }::name set)

shows (p · M) = M
  ⟨proof⟩

abbreviation
  outputJudge (-⟨-⟩ [110, 110] 110) where M⟨N⟩ ≡ M( $\nu^*(\square)$ )⟨N⟩

abbreviation
  brOutputJudge (i-⟨-⟩ [110, 110] 110) where  $iM$ ⟨N⟩ ≡  $iM$ ( $\nu^*(\square)$ )⟨N⟩

declare [[unify-trace-bound=100]]

locale env = substPsi substTerm substAssert substCond +
  assertion SCompose' SImp' SBottom' SChanEq' SOutCon' SInCon'
  for substTerm :: ('a::fs-name) ⇒ name list ⇒ 'a::fs-name list' ⇒ 'a'
    and substAssert :: ('b::fs-name) ⇒ name list ⇒ 'a::fs-name list' ⇒ 'b'
    and substCond :: ('c::fs-name) ⇒ name list ⇒ 'a::fs-name list' ⇒ 'c'
    and SCompose' :: 'b' ⇒ 'b' ⇒ 'b'
    and SImp' :: 'b' ⇒ 'c' ⇒ bool
    and SBottom' :: 'b'
    and SChanEq' :: 'a' ⇒ 'a' ⇒ 'c'
    and SOutCon' :: 'a' ⇒ 'a' ⇒ 'c'
    and SInCon' :: 'a' ⇒ 'a' ⇒ 'c'

begin
notation SCompose' (infixr ⊗ 90)
notation SImp' (- ⊢ - [85, 85] 85)
notation FrameImp (- ⊢F - [85, 85] 85)
abbreviation
  FBottomJudge (⊥F 90) where ⊥F ≡ (FAssert SBottom')
notation SChanEq' (- ↔ - [90, 90] 90)
notation SOutCon' (- ≤ - [90, 90] 90)
notation SInCon' (- ≥ - [90, 90] 90)
notation substTerm (-[::=] [100, 100, 100] 100)
notation subs (-[::=] [100, 100, 100] 100)
notation AssertionStatEq (- ≃ - [80, 80] 80)
notation FrameStatEq (- ≃F - [80, 80] 80)
notation SBottom' (1 190)
abbreviation insertAssertion' (insertAssertion) where insertAssertion' ≡ assertionAux.insertAssertion (⊗)

inductive semantics :: 'b' ⇒ ('a', 'b', 'c') psi ⇒ ('a', 'b', 'c') residual ⇒ bool

```

$(- \triangleright - \mapsto - [50, 50, 50] 50)$

where

cInput: $\llbracket \Psi \vdash M \leftrightarrow K; \text{distinct } xvec; \text{set } xvec \subseteq \text{supp } N; xvec \#* Tvec; \text{length } xvec = \text{length } Tvec; xvec \#* \Psi; xvec \#* M; xvec \#* K \rrbracket \Longrightarrow \Psi \triangleright M(\lambda * xvec \ N).P \mapsto K(\langle N[xvec::=Tvec] \rangle) \prec P[xvec::=Tvec]$

| *cBrInput*: $\llbracket \Psi \vdash K \succeq M; \text{distinct } xvec; \text{set } xvec \subseteq \text{supp } N; xvec \#* Tvec; \text{length } xvec = \text{length } Tvec; xvec \#* \Psi; xvec \#* M; xvec \#* K \rrbracket \Longrightarrow \Psi \triangleright M(\lambda * xvec \ N).P \mapsto \text{!}K(\langle N[xvec::=Tvec] \rangle) \prec P[xvec::=Tvec]$

| *Output*: $\llbracket \Psi \vdash M \leftrightarrow K \rrbracket \Longrightarrow \Psi \triangleright M\langle N \rangle.P \mapsto K\langle N \rangle \prec P$

| *BrOutput*: $\llbracket \Psi \vdash M \preceq K \rrbracket \Longrightarrow \Psi \triangleright M\langle N \rangle.P \mapsto \text{!}K\langle N \rangle \prec P$

| *Case*: $\llbracket \Psi \triangleright P \mapsto Rs; (\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \Longrightarrow \Psi \triangleright \text{Cases } Cs \mapsto Rs$

| *cPar1*: $\llbracket (\Psi \otimes \Psi_Q) \triangleright P \mapsto \alpha \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* \alpha; A_Q \#* P'; \text{distinct}(bn \ \alpha); bn \ \alpha \#* \Psi; bn \ \alpha \#* \Psi_Q; bn \ \alpha \#* Q; bn \ \alpha \#* P; bn \ \alpha \#* (\text{subject } \alpha) \rrbracket \Longrightarrow \Psi \triangleright P \parallel Q \mapsto \alpha \prec (P' \parallel Q)$

| *cPar2*: $\llbracket (\Psi \otimes \Psi_P) \triangleright Q \mapsto \alpha \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* \alpha; A_P \#* Q'; \text{distinct}(bn \ \alpha); bn \ \alpha \#* \Psi; bn \ \alpha \#* \Psi_P; bn \ \alpha \#* P; bn \ \alpha \#* Q; bn \ \alpha \#* (\text{subject } \alpha) \rrbracket \Longrightarrow \Psi \triangleright P \parallel Q \mapsto \alpha \prec (P \parallel Q')$

| *cComm1*: $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M\langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; \Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * xvec)\langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; \Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q'; A_Q \#* xvec; \text{distinct } xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#* Q; xvec \#* K \rrbracket \Longrightarrow \Psi \triangleright P \parallel Q \mapsto \tau \prec (\nu * xvec)\langle P' \parallel Q' \rangle$

| *cComm2*: $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; \Psi \otimes \Psi_P \triangleright Q \mapsto K\langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; \Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q'; A_Q \#* xvec; \text{distinct } xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#* Q; xvec \#* K \rrbracket \Longrightarrow \Psi \triangleright P \parallel Q \mapsto \tau \prec (\nu * xvec)\langle P' \parallel Q' \rangle$

| *cBrMerge*: $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \text{!}M\langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; \Psi \otimes \Psi_P \triangleright Q \mapsto \text{!}M\langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; \Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q'; A_Q \#* xvec; \text{distinct } xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#* Q; xvec \#* K \rrbracket \Longrightarrow \Psi \triangleright P \parallel Q \mapsto \text{!}M\langle N \rangle \prec Q'$

A_Q ;

$$\begin{aligned}
& A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; \\
& A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; \\
& A_P \#* M; A_Q \#* M; \\
& A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; \\
& A_Q \#* Q; A_Q \#* Q' \rceil \implies \\
& \Psi \triangleright P \parallel Q \mapsto \imath M(\langle N \rangle) \prec (P' \parallel Q')
\end{aligned}$$

| *cBrComm1*: $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(\langle N \rangle) \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
distinct A_P ;

$\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\langle \nu * \text{vec} \rangle) \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_Q ;

$$\begin{aligned}
& A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; \\
& A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{vec}; \\
& A_P \#* M; A_Q \#* M; \\
& A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; \\
& A_Q \#* Q; A_Q \#* Q'; A_Q \#* \text{vec}; \text{distinct } \text{vec}; \\
& \text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* \Psi_Q; \text{vec} \#* P; \\
& \text{vec} \#* Q; \text{vec} \#* M \rceil \implies \\
& \Psi \triangleright P \parallel Q \mapsto \imath M(\langle \nu * \text{vec} \rangle) \langle N \rangle \prec (P' \parallel Q')
\end{aligned}$$

| *cBrComm2*: $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(\langle \nu * \text{vec} \rangle) \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P,$
 $\Psi_P \rangle;$ *distinct* A_P ;

$\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\langle N \rangle) \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$ *distinct*
 A_Q ;

$$\begin{aligned}
& A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; \\
& A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{vec}; \\
& A_P \#* M; A_Q \#* M; \\
& A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; \\
& A_Q \#* Q; A_Q \#* Q'; A_Q \#* \text{vec}; \text{distinct } \text{vec}; \\
& \text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* \Psi_Q; \text{vec} \#* P; \\
& \text{vec} \#* Q; \text{vec} \#* M \rceil \implies \\
& \Psi \triangleright P \parallel Q \mapsto \imath M(\langle \nu * \text{vec} \rangle) \langle N \rangle \prec (P' \parallel Q')
\end{aligned}$$

| *cBrClose*: $\llbracket \Psi \triangleright P \mapsto \imath M(\langle \nu * \text{vec} \rangle) \langle N \rangle \prec P';$

$$\begin{aligned}
& x \in \text{supp } M; \\
& \text{distinct } \text{vec}; \text{vec} \#* \Psi; \text{vec} \#* P; \\
& \text{vec} \#* M; \\
& x \#* \Psi; x \#* \text{vec} \rceil \implies \\
& \Psi \triangleright (\nu x)P \mapsto \tau \prec (\nu x)(\langle \nu * \text{vec} \rangle P')
\end{aligned}$$

| *cOpen*: $\llbracket \Psi \triangleright P \mapsto M(\langle \nu * (\text{vec} @ \text{yvec}) \rangle) \langle N \rangle \prec P'; x \in \text{supp } N; x \#* \text{vec}; x \#*$
 $\text{yvec}; x \#* M; x \#* \Psi;$

distinct $\text{vec}; \text{distinct } \text{yvec};$
 $\text{vec} \#* \Psi; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* \text{yvec}; \text{yvec} \#* \Psi; \text{yvec} \#* P;$
 $\text{yvec} \#* M \rceil \implies$

$$\Psi \triangleright (\nu x)P \mapsto M(\langle \nu * (\text{vec} @ x \# \text{yvec}) \rangle) \langle N \rangle \prec P'$$

| *cBrOpen*: $\llbracket \Psi \triangleright P \mapsto \imath M(\langle \nu * (\text{vec} @ \text{yvec}) \rangle) \langle N \rangle \prec P'; x \in \text{supp } N; x \#* \text{vec};$
 $x \#* \text{yvec}; x \#* M; x \#* \Psi;$

distinct $\text{vec}; \text{distinct } \text{yvec};$
 $\text{vec} \#* \Psi; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* \text{yvec}; \text{yvec} \#* \Psi; \text{yvec} \#* P;$
 $\text{yvec} \#* M \rceil \implies$

$$\Psi \triangleright (\nu x)P \mapsto \mathfrak{i}M(\nu*(xvec@x\#yvec))\langle N \rangle \prec P'$$

| *cScope*: $\llbracket \Psi \triangleright P \mapsto \alpha \prec P'; x \# \Psi; x \# \alpha; bn \alpha \#* \Psi; bn \alpha \#* P; bn \alpha \#* (subject \alpha); distinct(bn \alpha) \rrbracket \Longrightarrow \Psi \triangleright (\nu x)P \mapsto \alpha \prec ((\nu x)P')$

| *Bang*: $\llbracket \Psi \triangleright P \parallel !P \mapsto Rs; guarded P \rrbracket \Longrightarrow \Psi \triangleright !P \mapsto Rs$

abbreviation

semanticsBottomJudge $(- \mapsto - [50, 50] 50)$ **where** $P \mapsto Rs \equiv \mathbf{1} \triangleright P \mapsto Rs$

equivariance *env.semantics*

nominal-inductive2 *env.semantics*

avoids *cInput*: *set xvec*

| *cBrInput*: *set xvec*

| *cPar1*: *set A_Q ∪ set(bn α)*

| *cPar2*: *set A_P ∪ set(bn α)*

| *cComm1*: *set A_P ∪ set A_Q ∪ set xvec*

| *cComm2*: *set A_P ∪ set A_Q ∪ set xvec*

| *cBrMerge*: *set A_P ∪ set A_Q*

| *cBrComm1*: *set A_P ∪ set A_Q ∪ set xvec*

| *cBrComm2*: *set A_P ∪ set A_Q ∪ set xvec*

| *cBrClose*: $\{x\} \cup set \ xvec$

| *cOpen*: $\{x\} \cup set \ xvec \cup set \ yvec$

| *cBrOpen*: $\{x\} \cup set \ xvec \cup set \ yvec$

| *cScope*: $\{x\} \cup set(bn \alpha)$

⟨*proof*⟩

lemma *nilTrans1*:

fixes $\Psi :: 'b$

and $M :: 'a$

and $xvec :: name \ list$

and $N :: 'a$

and $P :: ('a, 'b, 'c) \ psi$

assumes $\Psi \triangleright \mathbf{0} \mapsto M(\nu*xvec)\langle N \rangle \prec P$

shows *False*

⟨*proof*⟩

lemma *nilTrans1'*:

fixes $\Psi :: 'b$

and $M :: 'a$

and $xvec :: name \ list$

and $N :: 'a$

and $P :: ('a, 'b, 'c) \ psi$

assumes $\Psi \triangleright \mathbf{0} \mapsto \mathfrak{i}M(\nu*xvec)\langle N \rangle \prec P$

shows *False*

⟨*proof*⟩

lemma *nilTrans2*:
fixes Ψ :: 'b
and R_s :: ('a, 'b, 'c) *residual*

assumes $\Psi \triangleright \mathbf{0} \mapsto R_s$

shows *False*
 ⟨*proof*⟩

lemma *nilTrans3*:
fixes Ψ :: 'b
and M :: 'a
and M' :: 'a
and $xvec$:: *name list*
and $yvec$:: *name list*
and N :: 'a
and N' :: 'a
and P :: ('a, 'b, 'c) *psi*
and P' :: ('a, 'b, 'c) *psi*

assumes $\Psi \triangleright M(\lambda * xvec\ N).P \mapsto M'(\nu * yvec)\langle N' \rangle \prec P'$

shows *False*
 ⟨*proof*⟩

lemma *nilTrans3'*:
fixes Ψ :: 'b
and M :: 'a
and M' :: 'a
and $xvec$:: *name list*
and $yvec$:: *name list*
and N :: 'a
and N' :: 'a
and P :: ('a, 'b, 'c) *psi*
and P' :: ('a, 'b, 'c) *psi*

assumes $\Psi \triangleright M(\lambda * xvec\ N).P \mapsto ;M'(\nu * yvec)\langle N' \rangle \prec P'$

shows *False*
 ⟨*proof*⟩

lemma *nilTrans4*:
fixes Ψ :: 'b
and R_s :: ('a, 'b, 'c) *residual*

assumes $\Psi \triangleright M(\lambda * xvec\ N).P \mapsto \tau \prec P'$

shows *False*

<proof>

lemma *nilTrans5*:

fixes Ψ :: 'b
fixes Ψ' :: 'b
and M :: 'a
and $xvec$:: *name list*
and N :: 'a
and P :: ('a, 'b, 'c) *psi*

assumes $\Psi \triangleright \{\Psi'\} \mapsto M(\nu*xvec)\langle N \rangle \prec P$

shows *False*

<proof>

lemma *nilTrans5'*:

fixes Ψ :: 'b
fixes Ψ' :: 'b
and M :: 'a
and $xvec$:: *name list*
and N :: 'a
and P :: ('a, 'b, 'c) *psi*

assumes $\Psi \triangleright \{\Psi'\} \mapsto \text{!}M(\nu*xvec)\langle N \rangle \prec P$

shows *False*

<proof>

lemma *nilTrans6*:

fixes Ψ :: 'b
and R_s :: ('a, 'b, 'c) *residual*

assumes $\Psi \triangleright \{\Psi'\} \mapsto R_s$

shows *False*

<proof>

lemma *nilTrans[dest]*:

fixes Ψ :: 'b
and R_s :: ('a, 'b, 'c) *residual*
and M :: 'a
and $xvec$:: *name list*
and N :: 'a
and P :: ('a, 'b, 'c) *psi*
and K :: 'a
and $yvec$:: *name list*
and N' :: 'a
and P' :: ('a, 'b, 'c) *psi*
and CsP :: ('c \times ('a, 'b, 'c) *psi*) *list*

and $\Psi' :: 'b$

shows $\Psi \triangleright \mathbf{0} \mapsto Rs \implies False$
 and $\Psi \triangleright M(\lambda*xvec\ N).P \mapsto K(\nu*yvec)\langle N' \rangle \prec P' \implies False$
 and $\Psi \triangleright M(\lambda*xvec\ N).P \mapsto_i K(\nu*yvec)\langle N' \rangle \prec P' \implies False$
 and $\Psi \triangleright M(\lambda*xvec\ N).P \mapsto \tau \prec P' \implies False$
 and $\Psi \triangleright M\langle N \rangle.P \mapsto K\langle N' \rangle \prec P' \implies False$
 and $\Psi \triangleright M\langle N \rangle.P \mapsto_i K\langle N' \rangle \prec P' \implies False$
 and $\Psi \triangleright M\langle N \rangle.P \mapsto \tau \prec P' \implies False$
 and $\Psi \triangleright \{\Psi'\} \mapsto Rs \implies False$
 $\langle proof \rangle$

lemma *residualEq*:

fixes $\alpha :: 'a\ action$
 and $P :: ('a, 'b, 'c)\ psi$
 and $\beta :: 'a\ action$
 and $Q :: ('a, 'b, 'c)\ psi$

assumes $\alpha \prec P = \beta \prec Q$
 and $bn\ \alpha \#* (bn\ \beta)$
 and $distinct(bn\ \alpha)$
 and $distinct(bn\ \beta)$
 and $bn\ \alpha \#* (\alpha \prec P)$
 and $bn\ \beta \#* (\beta \prec Q)$

obtains p where $set\ p \subseteq set(bn\ \alpha) \times set(bn(p \cdot \alpha))$ and $distinctPerm\ p$ and $\beta = p \cdot \alpha$ and $Q = p \cdot P$ and $bn\ \alpha \#* \beta$ and $bn\ \alpha \#* Q$ and $bn(p \cdot \alpha) \#* \alpha$ and $bn(p \cdot \alpha) \#* P$
 $\langle proof \rangle$

lemma *semanticsInduct*[consumes 3, case-names *cAlpha cInput cBrInput cOutput cBrOutput cCase cPar1 cPar2 cComm1 cComm2 cBrMerge cBrComm1 cBrComm2 cBrClose cOpen cBrOpen cScope cBang*]:

fixes $\Psi :: 'b$
 and $P :: ('a, 'b, 'c)\ psi$
 and $\alpha :: 'a\ action$
 and $P' :: ('a, 'b, 'c)\ psi$
 and $Prop :: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c)\ psi \Rightarrow 'a\ action \Rightarrow ('a, 'b, 'c)\ psi \Rightarrow bool$
 and $C :: 'f::fs-name$

assumes $\Psi \triangleright P \mapsto \alpha \prec P'$
 and $bn\ \alpha \#* (subject\ \alpha)$
 and $distinct(bn\ \alpha)$
 and $rAlpha: \bigwedge \Psi\ P\ \alpha\ P'\ p\ C. \llbracket bn\ \alpha \#* \Psi; bn\ \alpha \#* P; bn\ \alpha \#* (subject\ \alpha); bn\ \alpha \#* C; bn\ \alpha \#* (bn(p \cdot \alpha)); set\ p \subseteq set(bn\ \alpha) \times set(bn(p \cdot \alpha)); distinctPerm\ p; (bn(p \cdot \alpha)) \#* \alpha; (bn(p \cdot \alpha)) \#* P'; Prop\ C\ \Psi\ P\ \alpha \rrbracket \implies$

$Prop\ C\ \Psi\ P\ (p \cdot \alpha)\ (p \cdot P')$

and $rInput: \bigwedge \Psi\ M\ K\ xvec\ N\ Tvec\ P\ C.$
 $\llbracket \Psi \vdash M \leftrightarrow K; \text{distinct } xvec; \text{set } xvec \subseteq \text{supp } N;$
 $\text{length } xvec = \text{length } Tvec; xvec \#* \Psi;$
 $xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (M(\lambda*xvec\ N).P)$
 $(K(\!(N[xvec::=Tvec]\!)))\ (P[xvec::=Tvec])$

and $rBrInput: \bigwedge \Psi\ K\ M\ xvec\ N\ Tvec\ P\ C.$
 $\llbracket \Psi \vdash K \succeq M; \text{distinct } xvec; \text{set } xvec \subseteq \text{supp } N;$
 $\text{length } xvec = \text{length } Tvec; xvec \#* \Psi;$
 $xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (M(\lambda*xvec\ N).P)$
 $(\!(K(\!(N[xvec::=Tvec]\!)))\ (P[xvec::=Tvec]))$

and $rOutput: \bigwedge \Psi\ M\ K\ N\ P\ C. \llbracket \Psi \vdash M \leftrightarrow K \rrbracket \implies Prop\ C\ \Psi\ (M\langle N \rangle.P)$
 $(K\langle N \rangle)\ P$

and $rBrOutput: \bigwedge \Psi\ M\ K\ N\ P\ C. \llbracket \Psi \vdash M \preceq K \rrbracket \implies Prop\ C\ \Psi\ (M\langle N \rangle.P)$
 $(\!(K\langle N \rangle)\ P)$

and $rCase: \bigwedge \Psi\ P\ \alpha\ P'\ \varphi\ Cs\ C. \llbracket \Psi \triangleright P \mapsto \alpha \prec P'; \bigwedge C. Prop\ C\ \Psi\ P\ \alpha\ P';$
 $(\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \implies$
 $Prop\ C\ \Psi\ (Cases\ Cs)\ \alpha\ P'$

and $rPar1: \bigwedge \Psi\ \Psi_Q\ P\ \alpha\ P'\ A_Q\ Q\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct}$
 $A_Q;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ \alpha\ P';$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* \alpha; A_Q \#* P'; A_Q \#* C;$
 $\text{distinct}(bn\ \alpha); bn\ \alpha \#* Q;$
 $bn\ \alpha \#* \Psi; bn\ \alpha \#* \Psi_Q; bn\ \alpha \#* P; bn\ \alpha \#* \text{subject } \alpha; bn\ \alpha \#* C \rrbracket$
 \implies
 $Prop\ C\ \Psi\ (P \parallel Q)\ \alpha\ (P' \parallel Q)$

and $rPar2: \bigwedge \Psi\ \Psi_P\ Q\ \alpha\ Q'\ A_P\ P\ C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct}$
 $A_P;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ \alpha\ Q';$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* \alpha; A_P \#* Q'; A_P \#* C;$
 $\text{distinct}(bn\ \alpha); bn\ \alpha \#* Q;$
 $bn\ \alpha \#* \Psi; bn\ \alpha \#* \Psi_P; bn\ \alpha \#* P; bn\ \alpha \#* \text{subject } \alpha; bn\ \alpha \#* C \rrbracket$
 \implies
 $Prop\ C\ \Psi\ (P \parallel Q)\ \alpha\ (P \parallel Q')$

and $rComm1: \bigwedge \Psi\ \Psi_Q\ P\ M\ N\ P'\ A_P\ \Psi_P\ Q\ K\ xvec\ Q'\ A_Q\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M\langle N \rangle \prec P'; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ (M\langle N \rangle)$
 $P';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu*xvec)\langle N \rangle \prec Q'; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q$
 $(K(\nu*xvec)\langle N \rangle)\ Q';$
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q';$

distinct xvec;
 $A_Q \#* \text{xvec}; \text{xvec} \#* \Psi; \text{xvec} \#* \Psi_P; \text{xvec} \#* \Psi_Q; \text{xvec} \#* P; \text{xvec} \#*$
 $M;$
 $\text{xvec} \#* Q; \text{xvec} \#* K; A_P \#* C; A_Q \#* C; \text{xvec} \#* C] \implies$
 $\text{Prop } C \Psi (P \parallel Q) (\tau) ((\nu*\text{xvec})(P' \parallel Q'))$
and $rComm2: \bigwedge \Psi \Psi_Q P M \text{xvec} N P' A_P \Psi_P Q K Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu*\text{xvec})(N) \prec P'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P$
 $(M(\nu*\text{xvec})(N)) P';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(N) \prec Q'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) Q (K(N))$
 $Q';$
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{xvec}; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q';$
distinct xvec;
 $A_Q \#* \text{xvec}; \text{xvec} \#* \Psi; \text{xvec} \#* \Psi_P; \text{xvec} \#* \Psi_Q; \text{xvec} \#* P; \text{xvec} \#*$
 $M;$
 $\text{xvec} \#* Q; \text{xvec} \#* K; A_P \#* C; A_Q \#* C; \text{xvec} \#* C] \implies$
 $\text{Prop } C \Psi (P \parallel Q) (\tau) ((\nu*\text{xvec})(P' \parallel Q'))$
and $rBrMerge: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto {}_iM(N) \prec P'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P$
 $({}_iM(N)) P';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto {}_iM(N) \prec Q'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) Q$
 $({}_iM(N)) Q';$
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C] \implies$
 $\text{Prop } C \Psi (P \parallel Q) ({}_iM(N)) (P' \parallel Q')$
and $rBrComm1: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q \text{xvec} Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto {}_iM(N) \prec P'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P ({}_iM(N))$
 $P';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto {}_iM(\nu*\text{xvec})(N) \prec Q'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P)$
 $Q ({}_iM(\nu*\text{xvec})(N)) Q';$
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{xvec}; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; \text{distinct } \text{xvec};$
 $A_P \#* M; A_Q \#* M; \text{xvec} \#* M;$
 $A_Q \#* \text{xvec}; \text{xvec} \#* \Psi; \text{xvec} \#* \Psi_P; \text{xvec} \#* \Psi_Q; \text{xvec} \#* P;$
 $\text{xvec} \#* Q; A_P \#* C; A_Q \#* C; \text{xvec} \#* C] \implies$
 $\text{Prop } C \Psi (P \parallel Q) ({}_iM(\nu*\text{xvec})(N)) (P' \parallel Q')$
and $rBrComm2: \bigwedge \Psi \Psi_Q P M \text{xvec} N P' A_P \Psi_P Q Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto {}_iM(\nu*\text{xvec})(N) \prec P'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q)$

$P \langle \text{iM}(\nu^*xvec)\langle N \rangle \rangle P'$;
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \text{iM}(\nu^*xvec)\langle N \rangle \prec Q'; \wedge C. \text{Prop } C \langle \Psi \otimes \Psi_P \rangle Q \langle \text{iM}(\nu^*xvec)\langle N \rangle \rangle$
 Q' ;
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $A_P \#^* \Psi; A_P \#^* \Psi_Q; A_P \#^* P; A_P \#^* N; A_P \#^* P';$
 $A_P \#^* Q; A_P \#^* Q'; A_P \#^* A_Q; A_P \#^* xvec; A_Q \#^* \Psi; A_Q \#^* \Psi_P;$
 $A_Q \#^* P; A_Q \#^* N; A_Q \#^* P'; A_Q \#^* Q; A_Q \#^* Q'; \text{distinct } xvec;$
 $A_P \#^* M; A_Q \#^* M; xvec \#^* M;$
 $A_Q \#^* xvec; xvec \#^* \Psi; xvec \#^* \Psi_P; xvec \#^* \Psi_Q; xvec \#^* P;$
 $xvec \#^* Q; A_P \#^* C; A_Q \#^* C; xvec \#^* C \implies$
 $\text{Prop } C \Psi (P \parallel Q) \langle \text{iM}(\nu^*xvec)\langle N \rangle \rangle (P' \parallel Q')$
and $rBrClose: \wedge \Psi P M xvec N P' x C.$
 $\llbracket \Psi \triangleright P \mapsto \text{iM}(\nu^*xvec)\langle N \rangle \prec P';$
 $\wedge C. \text{Prop } C \Psi P \langle \text{iM}(\nu^*xvec)\langle N \rangle \rangle P';$
 $x \in \text{supp } M;$
 $\text{distinct } xvec; xvec \#^* \Psi; xvec \#^* P;$
 $xvec \#^* M;$
 $x \#^* \Psi; x \#^* xvec \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x)P) (\tau) ((\nu x)(\nu^*xvec)P')$
and $rOpen: \wedge \Psi P M xvec yvec N P' x C.$
 $\llbracket \Psi \triangleright P \mapsto M(\nu^*(xvec@yvec))\langle N \rangle \prec P'; x \in \text{supp } N; \wedge C. \text{Prop } C$
 $\Psi P \langle M(\nu^*(xvec@yvec))\langle N \rangle \rangle P';$
 $x \#^* \Psi; x \#^* M; x \#^* xvec; x \#^* yvec; xvec \#^* \Psi; xvec \#^* P; xvec \#^* M;$
 $\text{distinct } xvec; \text{distinct } yvec;$
 $yvec \#^* \Psi; yvec \#^* P; yvec \#^* M; yvec \#^* C; x \#^* C; xvec \#^* C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x)P) \langle M(\nu^*(xvec@x\#yvec))\langle N \rangle \rangle P'$
and $rBrOpen: \wedge \Psi P M xvec yvec N P' x C.$
 $\llbracket \Psi \triangleright P \mapsto \text{iM}(\nu^*(xvec@yvec))\langle N \rangle \prec P'; x \in \text{supp } N; \wedge C. \text{Prop } C$
 $\Psi P \langle \text{iM}(\nu^*(xvec@yvec))\langle N \rangle \rangle P';$
 $x \#^* \Psi; x \#^* M; x \#^* xvec; x \#^* yvec; xvec \#^* \Psi; xvec \#^* P; xvec \#^* M;$
 $\text{distinct } xvec; \text{distinct } yvec;$
 $yvec \#^* \Psi; yvec \#^* P; yvec \#^* M; yvec \#^* C; x \#^* C; xvec \#^* C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x)P) \langle \text{iM}(\nu^*(xvec@x\#yvec))\langle N \rangle \rangle P'$
and $rScope: \wedge \Psi P \alpha P' x C.$
 $\llbracket \Psi \triangleright P \mapsto \alpha \prec P'; \wedge C. \text{Prop } C \Psi P \alpha P';$
 $x \#^* \Psi; x \#^* \alpha; bn \alpha \#^* \Psi;$
 $bn \alpha \#^* P; bn \alpha \#^* (\text{subject } \alpha); x \#^* C; bn \alpha \#^* C; \text{distinct}(bn \alpha) \rrbracket$
 \implies
 $\text{Prop } C \Psi ((\nu x)P) \alpha ((\nu x)P')$
and $rBang: \wedge \Psi P \alpha P' C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto \alpha \prec P'; \text{guarded } P; \wedge C. \text{Prop } C \Psi (P \parallel !P) \alpha$
 $P \rrbracket \implies$
 $\text{Prop } C \Psi (!P) \alpha P'$

shows $\text{Prop } C \Psi P \alpha P'$
 $\langle \text{proof} \rangle$

lemma $\text{outputInduct}[\text{consumes } 1, \text{case-names } cOutput \ cCase \ cPar1 \ cPar2 \ cOpen$

cScope cBang]:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c)$ *psi*
and M $:: 'a$
and B $:: ('a, 'b, 'c)$ *boundOutput*
and $Prop$ $:: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c)$ *psi* \Rightarrow
 $'a \Rightarrow ('a, 'b, 'c)$ *boundOutput* $\Rightarrow bool$
and C $:: 'f::fs-name$

assumes $\Psi \triangleright P \mapsto ROut M B$
and $rOutput$: $\bigwedge \Psi M K N P C. [\Psi \vdash M \leftrightarrow K] \Longrightarrow Prop C \Psi (M \langle N \rangle . P) K (N \prec' P)$
and $rCase$: $\bigwedge \Psi P M B \varphi Cs C.$
 $[\Psi \triangleright P \mapsto (ROut M B); \bigwedge C. Prop C \Psi P M B; (\varphi, P) \in set Cs;$
 $\Psi \vdash \varphi; guarded P] \Longrightarrow$
 $Prop C \Psi (Cases Cs) M B$
and $rPar1$: $\bigwedge \Psi \Psi_Q P M xvec N P' A_Q Q C.$
 $[\Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec) \langle N \rangle \prec P'; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle; distinct A_Q;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P M ((\nu * xvec) N \prec' P');$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M;$
 $A_Q \#* xvec; A_Q \#* N; A_Q \#* P'; A_Q \#* C; xvec \#* Q;$
 $xvec \#* \Psi; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#* C] \Longrightarrow$
 $Prop C \Psi (P \parallel Q) M ((\nu * xvec) N \prec' (P' \parallel Q))$
and $rPar2$: $\bigwedge \Psi \Psi_P Q M xvec N Q' A_P P C.$
 $[\Psi \otimes \Psi_P \triangleright Q \mapsto M(\nu * xvec) \langle N \rangle \prec Q'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle; distinct A_P;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q M ((\nu * xvec) N \prec' Q');$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M;$
 $A_P \#* xvec; A_P \#* N; A_P \#* Q'; A_P \#* C; xvec \#* P;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* Q; xvec \#* M; xvec \#* C] \Longrightarrow$
 $Prop C \Psi (P \parallel Q) M ((\nu * xvec) N \prec' (P \parallel Q'))$
and $rOpen$: $\bigwedge \Psi P M xvec yvec N P' x C.$
 $[\Psi \triangleright P \mapsto M(\nu * (xvec @ yvec)) \langle N \rangle \prec P'; x \in supp N; \bigwedge C. Prop C$
 $\Psi P M ((\nu * (xvec @ yvec)) N \prec' P');$
 $x \# \Psi; x \# M; x \# xvec; x \# yvec; xvec \# \Psi; xvec \# P; xvec \# M;$
 $xvec \# yvec; yvec \# \Psi; yvec \# P; yvec \# M; yvec \# C; x \# C;$
 $xvec \# C] \Longrightarrow$
 $Prop C \Psi ((\nu x) P) M ((\nu * (xvec @ x \# yvec)) N \prec' P')$
and $rScope$: $\bigwedge \Psi P M xvec N P' x C.$
 $[\Psi \triangleright P \mapsto M(\nu * xvec) \langle N \rangle \prec P'; \bigwedge C. Prop C \Psi P M ((\nu * xvec) N$
 $\prec' P');$
 $x \# \Psi; x \# M; x \# xvec; x \# N; xvec \# \Psi; xvec \# P; xvec \# M;$
 $x \# C; xvec \# C] \Longrightarrow$
 $Prop C \Psi ((\nu x) P) M ((\nu * xvec) N \prec' (\nu x) P')$
and $rBang$: $\bigwedge \Psi P M B C.$
 $[\Psi \triangleright P \parallel !P \mapsto (ROut M B); guarded P; \bigwedge C. Prop C \Psi (P \parallel$
 $!P) M B] \Longrightarrow$
 $Prop C \Psi (!P) M B$

shows $Prop\ C\ \Psi\ P\ M\ B$
 $\langle proof \rangle$

lemma $brOutputInduct[consumes\ 1,\ case-names\ cBrOutput\ cCase\ cPar1\ cPar2\ cBrComm1\ cBrComm2\ cBrOpen\ cScope\ cBang]$:

fixes Ψ $:: 'b$
and P $:: ('a,\ 'b,\ 'c)\ psi$
and M $:: 'a$
and B $:: ('a,\ 'b,\ 'c)\ boundOutput$
and $Prop$ $:: 'f::fs-name \Rightarrow 'b \Rightarrow ('a,\ 'b,\ 'c)\ psi \Rightarrow 'a \Rightarrow ('a,\ 'b,\ 'c)\ boundOutput \Rightarrow bool$
and C $:: 'f::fs-name$

assumes $\Psi \triangleright P \mapsto RBrOut\ M\ B$

and $rBrOutput: \bigwedge \Psi\ M\ K\ N\ P\ C. [\Psi \vdash M \preceq K] \Longrightarrow Prop\ C\ \Psi\ (M \langle N \rangle . P)\ K$
 $(N \prec' P)$

and $rCase: \bigwedge \Psi\ P\ M\ B\ \varphi\ Cs\ C.$

$[\Psi \triangleright P \mapsto (RBrOut\ M\ B); \bigwedge C. Prop\ C\ \Psi\ P\ M\ B; (\varphi,\ P) \in set\ Cs; \Psi \vdash \varphi; guarded\ P] \Longrightarrow$

$Prop\ C\ \Psi\ (Cases\ Cs)\ M\ B$

and $rPar1: \bigwedge \Psi\ \Psi_Q\ P\ M\ xvec\ N\ P'\ A_Q\ Q\ C.$

$[\Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P'; extractFrame\ Q = \langle A_Q,\ \Psi_Q \rangle; distinct\ A_Q;$

$\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ M\ ((\nu * xvec) N \prec' P');$

$A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M;$

$A_Q \#* xvec; A_Q \#* N; A_Q \#* P'; A_Q \#* C; xvec \#* Q;$

$xvec \#* \Psi; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#* C] \Longrightarrow$

$Prop\ C\ \Psi\ (P \parallel Q)\ M\ ((\nu * xvec) N \prec' (P' \parallel Q))$

and $rPar2: \bigwedge \Psi\ \Psi_P\ Q\ M\ xvec\ N\ Q'\ A_P\ P\ C.$

$[\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec Q'; extractFrame\ P = \langle A_P,\ \Psi_P \rangle; distinct\ A_P;$

$\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ M\ ((\nu * xvec) N \prec' Q');$

$A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M;$

$A_P \#* xvec; A_P \#* N; A_P \#* Q'; A_P \#* C; xvec \#* P;$

$xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* Q; xvec \#* M; xvec \#* C] \Longrightarrow$

$Prop\ C\ \Psi\ (P \parallel Q)\ M\ ((\nu * xvec) N \prec' (P \parallel Q'))$

and $rBrComm1: \bigwedge \Psi\ \Psi_Q\ P\ M\ N\ P'\ A_P\ \Psi_P\ Q\ xvec\ Q'\ A_Q\ C.$

$[\Psi \otimes \Psi_Q \triangleright P \mapsto_i M \langle N \rangle \prec P'; extractFrame\ P = \langle A_P,\ \Psi_P \rangle; distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec Q'; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)$
 $Q\ M\ ((\nu * xvec) N \prec' Q');$

$extractFrame\ Q = \langle A_Q,\ \Psi_Q \rangle; distinct\ A_Q;$

$A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$

$A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$

$A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; distinct\ xvec;$

$A_P \#* M; A_Q \#* M; xvec \#* M;$

$A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$

$xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C] \Longrightarrow$

$Prop\ C\ \Psi\ (P \parallel Q)\ M\ ((\nu * xvec) N \prec' (P' \parallel Q'))$

and *rBrComm2*: $\bigwedge \Psi \Psi_Q P M \text{vec} N P' A_P \Psi_P Q Q' A_Q C$.
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu^* \text{vec}) \langle N \rangle \prec P'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P M ((\nu^* \text{vec}) N \prec' P') \rrbracket$;
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P$;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M \langle N \rangle \prec Q'$;
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q$;
 $A_P \#^* \Psi; A_P \#^* \Psi_Q; A_P \#^* P; A_P \#^* N; A_P \#^* P'$;
 $A_P \#^* Q; A_P \#^* Q'; A_P \#^* A_Q; A_P \#^* \text{vec}; A_Q \#^* \Psi; A_Q \#^* \Psi_P$;
 $A_Q \#^* P; A_Q \#^* N; A_Q \#^* P'; A_Q \#^* Q; A_Q \#^* Q'; \text{distinct } \text{vec}$;
 $A_P \#^* M; A_Q \#^* M; \text{vec} \#^* M$;
 $A_Q \#^* \text{vec}; \text{vec} \#^* \Psi; \text{vec} \#^* \Psi_P; \text{vec} \#^* \Psi_Q; \text{vec} \#^* P$;
 $\text{vec} \#^* Q; A_P \#^* C; A_Q \#^* C; \text{vec} \#^* C \rrbracket \implies$
 $\text{Prop } C \Psi (P \parallel Q) M ((\nu^* \text{vec}) N \prec' (P' \parallel Q'))$

and *rBrOpen*: $\bigwedge \Psi P M \text{vec} \text{yvec} N P' x C$.
 $\llbracket \Psi \triangleright P \mapsto_i M(\nu^*(\text{vec} @ \text{yvec})) \langle N \rangle \prec P'; x \in \text{supp } N; \bigwedge C. \text{Prop } C \Psi P M ((\nu^*(\text{vec} @ \text{yvec})) N \prec' P') \rrbracket$;
 $x \#^* \Psi; x \#^* M; x \#^* \text{vec}; x \#^* \text{yvec}; \text{vec} \#^* \Psi; \text{vec} \#^* P; \text{vec} \#^* M$;
 $\text{vec} \#^* \text{yvec}; \text{yvec} \#^* \Psi; \text{yvec} \#^* P; \text{yvec} \#^* M; \text{yvec} \#^* C; x \#^* C$;
 $\text{vec} \#^* C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x) P) M ((\nu^*(\text{vec} @ x \# \text{yvec})) N \prec' P')$

and *rScope*: $\bigwedge \Psi P M \text{vec} N P' x C$.
 $\llbracket \Psi \triangleright P \mapsto_i M(\nu^* \text{vec}) \langle N \rangle \prec P'; \bigwedge C. \text{Prop } C \Psi P M ((\nu^* \text{vec}) N \prec' P') \rrbracket$;
 $x \#^* \Psi; x \#^* M; x \#^* \text{vec}; x \#^* N; \text{vec} \#^* \Psi; \text{vec} \#^* P; \text{vec} \#^* M$;
 $x \#^* C; \text{vec} \#^* C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x) P) M ((\nu^* \text{vec}) N \prec' (\nu x) P')$

and *rBang*: $\bigwedge \Psi P M B C$.
 $\llbracket \Psi \triangleright P \parallel !P \mapsto (RBrOut M B); \text{guarded } P; \bigwedge C. \text{Prop } C \Psi (P \parallel !P) M B \rrbracket \implies$
 $\text{Prop } C \Psi (!P) M B$

shows $\text{Prop } C \Psi P M B$
 $\langle \text{proof} \rangle$

lemma *boundOutputBindObject*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $M :: 'a$
and $\text{yvec} :: \text{name list}$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{psi}$
and $y :: \text{name}$

assumes $\Psi \triangleright P \mapsto_\alpha \prec P'$

and $\text{bn } \alpha \#^* \text{subject } \alpha$
and $\text{distinct}(\text{bn } \alpha)$
and $y \in \text{set}(\text{bn } \alpha)$

shows $y \in \text{supp}(\text{object } \alpha)$

$\langle \text{proof} \rangle$

lemma *alphaBoundOutputChain'*:
fixes *yvec* :: *name list*
and *xvec* :: *name list*
and *B* :: ('a, 'b, 'c) *boundOutput*

assumes *length xvec = length yvec*
and *yvec* #* *B*
and *yvec* #* *xvec*
and *distinct yvec*

shows $(\nu * xvec)B = (\nu * yvec)([xvec\ yvec] \cdot_v B)$
<proof>

lemma *alphaBoundOutputChain''*:
fixes *yvec* :: *name list*
and *xvec* :: *name list*
and *N* :: 'a
and *P* :: ('a, 'b, 'c) *psi*

assumes *length xvec = length yvec*
and *yvec* #* *N*
and *yvec* #* *P*
and *yvec* #* *xvec*
and *distinct yvec*

shows $(\nu * xvec)(N \prec' P) = (\nu * yvec)(([xvec\ yvec] \cdot_v N) \prec' ([xvec\ yvec] \cdot_v P))$
<proof>

lemma *alphaDistinct*:
fixes *xvec* :: *name list*
and *N* :: 'a
and *P* :: ('a, 'b, 'c) *psi*
and *yvec* :: *name list*
and *M* :: 'a
and *Q* :: ('a, 'b, 'c) *psi*

assumes $\alpha \prec P = \beta \prec Q$
and *distinct(bn* $\alpha)$
and $\bigwedge x. x \in \text{set}(\text{bn } \alpha) \implies x \in \text{supp}(\text{object } \alpha)$
and *bn* α #* *bn* β
and *bn* α #* (*object* β)
and *bn* α #* *Q*

shows *distinct(bn* $\beta)$
<proof>

lemma *boundOutputDistinct*:
fixes Ψ :: 'b

and P :: ('a, 'b, 'c) psi
and α :: 'a action
and P' :: ('a, 'b, 'c) psi

assumes $\Psi \triangleright P \mapsto \alpha \prec P'$

shows *distinct*(bn α)
 ⟨proof⟩

lemma *inputDistinct*:

fixes Ψ :: 'b
and M :: 'a
and $xvec$:: name list
and N :: 'a
and P :: ('a, 'b, 'c) psi
and Rs :: ('a, 'b, 'c) residual

assumes $\Psi \triangleright M(\lambda * xvec N).P \mapsto Rs$

shows *distinct* $xvec$
 ⟨proof⟩

lemma *outputInduct'*[*consumes 2, case-names cAlpha cOutput cCase cPar1 cPar2 cOpen cScope cBang*]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $yvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and $Prop$:: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow
 'a \Rightarrow name list \Rightarrow 'a \Rightarrow ('a, 'b, 'c) psi \Rightarrow bool
and C :: 'f::fs-name

assumes $\Psi \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'$

and $xvec \#* M$
and $rAlpha$: $\bigwedge \Psi P M xvec N P' p C. \llbracket xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* C; xvec \#* (p \cdot xvec) \rrbracket$;

$set\ p \subseteq set\ xvec \times set(p \cdot xvec); distinctPerm\ p;$
 $(p \cdot xvec) \#* N; (p \cdot xvec) \#* P'; Prop\ C\ \Psi\ P\ M$

$xvec\ N\ P' \rrbracket \Longrightarrow$

$Prop\ C\ \Psi\ P\ M\ (p \cdot xvec)\ (p \cdot N)\ (p \cdot P')$

and $rOutput$: $\bigwedge \Psi M K N P C. \llbracket \Psi \vdash M \leftrightarrow K \rrbracket \Longrightarrow Prop\ C\ \Psi\ (M\langle N \rangle.P)\ K\ (\llbracket N\ P \rrbracket)$

and $rCase$: $\bigwedge \Psi P M xvec N P' \varphi Cs C. \llbracket \Psi \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'; \bigwedge C. Prop\ C\ \Psi\ P\ M\ xvec\ N\ P'; (\varphi, P) \in set\ Cs; \Psi \vdash \varphi; guarded\ P \rrbracket \Longrightarrow$

$Prop\ C\ \Psi\ (Cases\ Cs)\ M\ xvec\ N\ P'$

and $rPar1$: $\bigwedge \Psi \Psi_Q P M xvec N P' A_Q Q C.$

$\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'; extractFrame\ Q = \langle A_Q, \rrbracket$

Ψ_Q); *distinct* A_Q ;
 $\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P M \text{vec } N P'$;
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M$;
 $A_Q \#* \text{vec}; A_Q \#* N; A_Q \#* P'; A_Q \#* C; \text{vec} \#* Q$;
 $\text{vec} \#* \Psi; \text{vec} \#* \Psi_Q; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* C \implies$
 $\text{Prop } C \Psi (P \parallel Q) M \text{vec } N (P' \parallel Q)$
and *rPar2*: $\bigwedge \Psi \Psi_P Q M \text{vec } N Q' A_P P C$.
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto M(\nu * \text{vec}) \langle N \rangle \prec Q'; \text{extractFrame } P = \langle A_P,$
 $\Psi_P \rangle$; *distinct* A_P ;
 $\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) Q M \text{vec } N Q'$;
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M$;
 $A_P \#* \text{vec}; A_P \#* N; A_P \#* Q'; A_P \#* C; \text{vec} \#* Q$;
 $\text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* C \implies$
 $\text{Prop } C \Psi (P \parallel Q) M \text{vec } N (P \parallel Q')$
and *rOpen*: $\bigwedge \Psi P M \text{vec } yvec N P' x C$.
 $\llbracket \Psi \triangleright P \mapsto M(\nu * (\text{vec} @ yvec)) \langle N \rangle \prec P'; x \in \text{supp } N; \bigwedge C. \text{Prop } C$
 $\Psi P M (\text{vec} @ yvec) N P'$;
 $x \# \Psi; x \# M; x \# \text{vec}; x \# yvec; \text{vec} \#* \Psi; \text{vec} \#* P; \text{vec} \#* M$;
 $yvec \#* \Psi; yvec \#* P; yvec \#* M; yvec \#* C; x \# C; \text{vec} \#* C \implies$
 $\text{Prop } C \Psi ((\nu x)P) M (\text{vec} @ x \# yvec) N P'$
and *rScope*: $\bigwedge \Psi P M \text{vec } N P' x C$.
 $\llbracket \Psi \triangleright P \mapsto M(\nu * \text{vec}) \langle N \rangle \prec P'; \bigwedge C. \text{Prop } C \Psi P M \text{vec } N P'$;
 $x \# \Psi; x \# M; x \# \text{vec}; x \# N; \text{vec} \#* \Psi$;
 $\text{vec} \#* P; \text{vec} \#* M; x \# C; \text{vec} \#* C \implies$
 $\text{Prop } C \Psi ((\nu x)P) M \text{vec } N ((\nu x)P')$
and *rBang*: $\bigwedge \Psi P M \text{vec } N P' C$.
 $\llbracket \Psi \triangleright P \parallel !P \mapsto M(\nu * \text{vec}) \langle N \rangle \prec P'; \text{guarded } P; \bigwedge C. \text{Prop } C$
 $\Psi (P \parallel !P) M \text{vec } N P' \rrbracket \implies$
 $\text{Prop } C \Psi (!P) M \text{vec } N P'$
shows $\text{Prop } C \Psi P M \text{vec } N P'$
 $\langle \text{proof} \rangle$

lemma *brOutputInduct'*[*consumes 2, case-names cAlpha cBrOutput cCase cPar1 cPar2 cBrComm1 cBrComm2 cBrOpen cScope cBang*]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) *psi*
and M :: 'a
and *yvec* :: *name list*
and N :: 'a
and P' :: ('a, 'b, 'c) *psi*
and *Prop* :: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) *psi* \Rightarrow
' a \Rightarrow *name list* \Rightarrow 'a \Rightarrow ('a, 'b, 'c) *psi* \Rightarrow *bool*
and C :: 'f::fs-name

assumes $\Psi \triangleright P \mapsto_i M(\nu * \text{vec}) \langle N \rangle \prec P'$

and $\text{vec} \#* M$
and *rAlpha*: $\bigwedge \Psi P M \text{vec } N P' p C. \llbracket \text{vec} \#* \Psi; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* C; \text{vec} \#* (p \cdot \text{vec});$

$\text{set } p \subseteq \text{set } \text{vec} \times \text{set}(p \cdot \text{vec}); \text{distinctPerm } p;$

$$\begin{aligned}
& (p \cdot \text{vec}) \#* N; (p \cdot \text{vec}) \#* P'; \text{Prop } C \Psi P M \\
\text{vec } N P \llbracket \implies & \\
& \text{Prop } C \Psi P M (p \cdot \text{vec}) (p \cdot N) (p \cdot P') \\
\text{and } rBrOutput: \bigwedge \Psi M K N P C. \llbracket \Psi \vdash M \preceq K \rrbracket \implies & \text{Prop } C \Psi (M \langle N \rangle . P) K \\
(\parallel) N P & \\
\text{and } rCase: \bigwedge \Psi P M \text{vec } N P' \varphi Cs C. \llbracket \Psi \triangleright P \mapsto_i M(\nu * \text{vec}) \langle N \rangle \prec P'; & \\
\bigwedge C. \text{Prop } C \Psi P M \text{vec } N P'; (\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \implies & \\
& \text{Prop } C \Psi (\text{Cases } Cs) M \text{vec } N P' \\
\text{and } rPar1: \bigwedge \Psi \Psi_Q P M \text{vec } N P' A_Q Q C. & \\
\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu * \text{vec}) \langle N \rangle \prec P'; \text{extractFrame } Q = \langle A_Q, & \\
\Psi_Q \rangle; \text{distinct } A_Q; & \\
\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P M \text{vec } N P'; & \\
A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; & \\
A_Q \#* \text{vec}; A_Q \#* N; A_Q \#* P'; A_Q \#* C; \text{vec} \#* Q; & \\
\text{vec} \#* \Psi; \text{vec} \#* \Psi_Q; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* C \rrbracket \implies & \\
& \text{Prop } C \Psi (P \parallel Q) M \text{vec } N (P' \parallel Q) \\
\text{and } rPar2: \bigwedge \Psi \Psi_P Q M \text{vec } N Q' A_P P C. & \\
\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * \text{vec}) \langle N \rangle \prec Q'; \text{extractFrame } P = \langle A_P, & \\
\Psi_P \rangle; \text{distinct } A_P; & \\
\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) Q M \text{vec } N Q'; & \\
A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; & \\
A_P \#* \text{vec}; A_P \#* N; A_P \#* Q'; A_P \#* C; \text{vec} \#* Q; & \\
\text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#* C \rrbracket \implies & \\
& \text{Prop } C \Psi (P \parallel Q) M \text{vec } N (P' \parallel Q') \\
\text{and } rBrComm1: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q \text{vec } Q' A_Q C. & \\
\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M \langle N \rangle \prec P'; & \\
\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; & \\
\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * \text{vec}) \langle N \rangle \prec Q'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) & \\
Q M \text{vec } N Q'; & \\
\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; & \\
A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; & \\
A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{vec}; A_Q \#* \Psi; A_Q \#* \Psi_P; & \\
A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; \text{distinct } \text{vec}; & \\
A_P \#* M; A_Q \#* M; \text{vec} \#* M; & \\
A_Q \#* \text{vec}; \text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* \Psi_Q; \text{vec} \#* P; & \\
\text{vec} \#* Q; A_P \#* C; A_Q \#* C; \text{vec} \#* C \rrbracket \implies & \\
& \text{Prop } C \Psi (P \parallel Q) M \text{vec } N (P' \parallel Q') \\
\text{and } rBrComm2: \bigwedge \Psi \Psi_Q P M \text{vec } N P' A_P \Psi_P Q Q' A_Q C. & \\
\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu * \text{vec}) \langle N \rangle \prec P'; \bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) & \\
P M \text{vec } N P'; & \\
\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; & \\
\Psi \otimes \Psi_P \triangleright Q \mapsto_i M \langle N \rangle \prec Q'; & \\
\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; & \\
A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; & \\
A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{vec}; A_Q \#* \Psi; A_Q \#* \Psi_P; & \\
A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; \text{distinct } \text{vec}; & \\
A_P \#* M; A_Q \#* M; \text{vec} \#* M; & \\
A_Q \#* \text{vec}; \text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* \Psi_Q; \text{vec} \#* P; & \\
\text{vec} \#* Q; A_P \#* C; A_Q \#* C; \text{vec} \#* C \rrbracket \implies & \\
& \text{Prop } C \Psi (P \parallel Q) M \text{vec } N (P' \parallel Q')
\end{aligned}$$

$$\text{Prop } C \Psi (P \parallel Q) M \text{ xvec } N (P' \parallel Q')$$
and $rOpen$: $\bigwedge \Psi P M \text{ xvec } yvec N P' x C.$
 $\llbracket \Psi \triangleright P \mapsto_i M(\nu*(\text{xvec}@yvec)) \langle N \rangle \prec P'; x \in \text{supp } N; \bigwedge C. \text{Prop } C \Psi P M (\text{xvec}@yvec) N P';$
 $x \# \Psi; x \# M; x \# \text{xvec}; x \# yvec; \text{xvec} \#* \Psi; \text{xvec} \#* P; \text{xvec} \#* M;$
 $yvec \#* \Psi; yvec \#* P; yvec \#* M; yvec \#* C; x \# C; \text{xvec} \#* C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x)P) M (\text{xvec}@x\#yvec) N P'$
and $rScope$: $\bigwedge \Psi P M \text{ xvec } N P' x C.$
 $\llbracket \Psi \triangleright P \mapsto_i M(\nu*\text{xvec}) \langle N \rangle \prec P'; \bigwedge C. \text{Prop } C \Psi P M \text{ xvec } N P';$
 $x \# \Psi; x \# M; x \# \text{xvec}; x \# N; \text{xvec} \#* \Psi;$
 $\text{xvec} \#* P; \text{xvec} \#* M; x \# C; \text{xvec} \#* C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x)P) M \text{ xvec } N ((\nu x)P')$
and $rBang$: $\bigwedge \Psi P M \text{ xvec } N P' C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto_i M(\nu*\text{xvec}) \langle N \rangle \prec P'; \text{guarded } P; \bigwedge C. \text{Prop } C$
 $\Psi (P \parallel !P) M \text{ xvec } N P' \rrbracket \implies$
 $\text{Prop } C \Psi (!P) M \text{ xvec } N P'$
shows $\text{Prop } C \Psi P M \text{ xvec } N P'$
 $\langle \text{proof} \rangle$

lemma $\text{inputInduct}[\text{consumes } 1, \text{case-names } cInput \text{ cCase } cPar1 \text{ cPar2 } cScope \text{ cBang}]$:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) \text{psi}$
and M $:: 'a$
and N $:: 'a$
and P' $:: ('a, 'b, 'c) \text{psi}$
and $\text{Prop} :: 'f::\text{fs-name} \Rightarrow 'b \Rightarrow ('a, 'b, 'c) \text{psi} \Rightarrow$
 $'a \Rightarrow 'a \Rightarrow ('a, 'b, 'c) \text{psi} \Rightarrow \text{bool}$
and C $:: 'f::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright P \mapsto_i M \langle N \rangle \prec P'$
and $rInput$: $\bigwedge \Psi M K \text{ xvec } N \text{ Tvec } P C.$
 $\llbracket \Psi \vdash M \leftrightarrow K; \text{distinct } \text{xvec}; \text{set } \text{xvec} \subseteq \text{supp } N;$
 $\text{length } \text{xvec} = \text{length } \text{Tvec}; \text{xvec} \#* \Psi;$
 $\text{xvec} \#* M; \text{xvec} \#* K; \text{xvec} \#* C \rrbracket \implies$
 $\text{Prop } C \Psi (M(\lambda*\text{xvec } N).P)$
 $K (N[\text{xvec}::=\text{Tvec}]) (P[\text{xvec}::=\text{Tvec}])$
and $rCase$: $\bigwedge \Psi P M N P' \varphi Cs C. \llbracket \Psi \triangleright P \mapsto_i M \langle N \rangle \prec P'; \bigwedge C. \text{Prop } C \Psi$
 $P M N P'; (\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \implies$
 $\text{Prop } C \Psi (\text{Cases } Cs) M N P'$
and $rPar1$: $\bigwedge \Psi \Psi_Q P M N P' A_Q Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M \langle N \rangle \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
 $\text{distinct } A_Q;$
 $\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P M N P'; \text{distinct } A_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* N;$
 $A_Q \#* P'; A_Q \#* C \rrbracket \implies$
 $\text{Prop } C \Psi (P \parallel Q) M N (P' \parallel Q)$
and $rPar2$: $\bigwedge \Psi \Psi_P Q M N Q' A_P P C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M \langle N \rangle \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$

distinct A_P ;
 $\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) Q M N Q'; \text{distinct } A_P$;
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* N$;
 $A_P \#* Q'; A_P \#* C \implies$
 $\text{Prop } C \Psi (P \parallel Q) M N (P \parallel Q')$
and $rScope: \bigwedge \Psi P M N P' x C.$
 $\llbracket \Psi \triangleright P \mapsto M(N) \prec P'; \bigwedge C. \text{Prop } C \Psi P M N P'; x \# \Psi; x \#$
 $M; x \# N; x \# C \rrbracket \implies$
 $\text{Prop } C \Psi ((\nu x)P) M N ((\nu x)P')$
and $rBang: \bigwedge \Psi P M N P' C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto M(N) \prec P'; \text{guarded } P; \bigwedge C. \text{Prop } C \Psi (P \parallel$
 $!P) M N P' \rrbracket \implies \text{Prop } C \Psi (!P) M N P'$
shows $\text{Prop } C \Psi P M N P'$
 $\langle \text{proof} \rangle$

lemma $brInputInduct[\text{consumes } 1, \text{case-names } cBrInput \text{ cCase } cPar1 \text{ cPar2 } cBrMerge \text{ cScope } cBang]$:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{psi}$
and $\text{Prop} :: 'f::\text{fs-name} \Rightarrow 'b \Rightarrow ('a, 'b, 'c) \text{psi} \Rightarrow$
 $'a \Rightarrow 'a \Rightarrow ('a, 'b, 'c) \text{psi} \Rightarrow \text{bool}$
and $C :: 'f::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright P \mapsto_i M(N) \prec P'$
and $rBrInput: \bigwedge \Psi K M \text{vec } N \text{Tvec } P C.$
 $\llbracket \Psi \vdash K \succeq M; \text{distinct } \text{vec}; \text{set } \text{vec} \subseteq \text{supp } N;$
 $\text{length } \text{vec} = \text{length } \text{Tvec}; \text{vec} \#* \Psi;$
 $\text{vec} \#* M; \text{vec} \#* K; \text{vec} \#* C \rrbracket \implies$
 $\text{Prop } C \Psi (M(\lambda * \text{vec } N).P)$
 $K (N[\text{vec}::=\text{Tvec}]) (P[\text{vec}::=\text{Tvec}])$
and $rCase: \bigwedge \Psi P M N P' \varphi Cs C. \llbracket \Psi \triangleright P \mapsto_i M(N) \prec P'; \bigwedge C. \text{Prop } C \Psi$
 $P M N P'; (\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \implies$
 $\text{Prop } C \Psi (\text{Cases } Cs) M N P'$
and $rPar1: \bigwedge \Psi \Psi_Q P M N P' A_Q Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_Q ;
 $\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_Q) P M N P'; \text{distinct } A_Q$;
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* N$;
 $A_Q \#* P'; A_Q \#* C \implies$
 $\text{Prop } C \Psi (P \parallel Q) M N (P' \parallel Q)$
and $rPar2: \bigwedge \Psi \Psi_P Q M N Q' A_P P C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M(N) \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
distinct A_P ;
 $\bigwedge C. \text{Prop } C (\Psi \otimes \Psi_P) Q M N Q'; \text{distinct } A_P$;
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* N$;
 $A_P \#* Q'; A_P \#* C \implies$

$Prop\ C\ \Psi\ (P\ \parallel\ Q)\ M\ N\ (P\ \parallel\ Q')$
and $rBrMerge: \bigwedge \Psi\ \Psi_Q\ P\ M\ N\ P'\ A_P\ \Psi_P\ Q\ Q'\ A_Q\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \iota M(N) \prec P'; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ M\ N$
 $P';$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \iota M(N) \prec Q'; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ M\ N$
 $Q';$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M \rrbracket \implies$
 $Prop\ C\ \Psi\ (P\ \parallel\ Q)\ M\ N\ (P' \parallel Q')$
and $rScope: \bigwedge \Psi\ P\ M\ N\ P'\ x\ C.$
 $\llbracket \Psi \triangleright P \mapsto \iota M(N) \prec P'; \bigwedge C. Prop\ C\ \Psi\ P\ M\ N\ P'; x \# \Psi; x \#$
 $M; x \# N; x \# C \rrbracket \implies$
 $Prop\ C\ \Psi\ ((\nu x)P)\ M\ N\ ((\nu x)P')$
and $rBang: \bigwedge \Psi\ P\ M\ N\ P'\ C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto \iota M(N) \prec P'; guarded\ P; \bigwedge C. Prop\ C\ \Psi\ (P \parallel$
 $!P)\ M\ N\ P' \rrbracket \implies Prop\ C\ \Psi\ (!P)\ M\ N\ P'$
shows $Prop\ C\ \Psi\ P\ M\ N\ P'$
 $\langle proof \rangle$

lemma $tauInduct[consumes\ 1, case-names\ cCase\ cPar1\ cPar2\ cComm1\ cComm2$
 $cBrClose\ cScope\ cBang]:$

fixes $\Psi \quad :: 'b$
and $P \quad :: ('a, 'b, 'c)\ psi$
and $Rs \quad :: ('a, 'b, 'c)\ residual$
and $Prop \quad :: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c)\ psi \Rightarrow$
 $('a, 'b, 'c)\ psi \Rightarrow bool$
and $C \quad :: 'f::fs-name$

assumes $Trans: \Psi \triangleright P \mapsto \tau \prec P'$

and $rCase: \bigwedge \Psi\ P\ P'\ \varphi\ Cs\ C. \llbracket \Psi \triangleright P \mapsto \tau \prec P'; \bigwedge C. Prop\ C\ \Psi\ P\ P'; (\varphi,$
 $P) \in set\ Cs; \Psi \vdash \varphi; guarded\ P \rrbracket \implies$

$Prop\ C\ \Psi\ (Cases\ Cs)\ P'$

and $rPar1: \bigwedge \Psi\ \Psi_Q\ P\ P'\ A_Q\ Q\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \tau \prec P'; extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct$
 $A_Q;$

$\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ P';$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi;$
 $A_Q \#* P'; A_Q \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (P \parallel Q)\ (P' \parallel Q)$

and $rPar2: \bigwedge \Psi\ \Psi_P\ Q\ Q'\ A_P\ P\ C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \tau \prec Q'; extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct$
 $A_P;$

$\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ Q';$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi;$

$A_P \#* Q'; A_P \#* C \Longrightarrow$
 $Prop C \Psi (P \parallel Q) (P \parallel Q')$

and $rComm1: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q K xvec Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu N) \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
distinct $A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu xvec)\langle N \rangle \prec Q'; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle;$ *distinct* $A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#*$
 $M;$
 $xvec \#* Q; xvec \#* K; A_P \#* C; A_Q \#* C; xvec \#* C \Longrightarrow$
 $Prop C \Psi (P \parallel Q) ((\nu xvec)(P' \parallel Q'))$

and $rComm2: \bigwedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q K Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle;$ *distinct* $A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(N) \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
distinct $A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#*$
 $M;$
 $xvec \#* Q; xvec \#* K; A_P \#* C; A_Q \#* C; xvec \#* C \Longrightarrow$
 $Prop C \Psi (P \parallel Q) ((\nu xvec)(P' \parallel Q'))$

and $rBrClose: \bigwedge \Psi P M xvec N P' A_P \Psi_P x C.$
 $\llbracket \Psi \triangleright P \mapsto {}_i M(\nu xvec)\langle N \rangle \prec P';$
 $x \in supp M;$
distinct $xvec; xvec \#* \Psi; xvec \#* P;$
 $xvec \#* M;$
 $x \#* \Psi; x \#* xvec \Longrightarrow$
 $Prop C \Psi ((\nu x)P) ((\nu x)((\nu xvec)P'))$

and $rScope: \bigwedge \Psi P P' x C.$
 $\llbracket \Psi \triangleright P \mapsto \tau \prec P'; \bigwedge C. Prop C \Psi P P'; x \#* \Psi; x \#* C \Longrightarrow$
 $Prop C \Psi ((\nu x)P) ((\nu x)P')$

and $rBang: \bigwedge \Psi P P' C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto \tau \prec P'; guarded P; \bigwedge C. Prop C \Psi (P \parallel !P) P \rrbracket$
 $\Longrightarrow Prop C \Psi (!P) P'$
shows $Prop C \Psi P P'$
(proof)

lemma *semanticsFrameInduct*[*consumes 3, case-names cAlpha cInput cBrInput*
cOutput cBrOutput cCase cPar1 cPar2 cComm1 cComm2 cBrMerge cBrComm1
cBrComm2 cBrClose cOpen cBrOpen cScope cBang]:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) psi$

and R_s :: ('a, 'b, 'c) residual
and A_P :: name list
and Ψ_P :: 'b
and $Prop$:: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow
('a, 'b, 'c) residual \Rightarrow name list \Rightarrow 'b \Rightarrow bool
and C :: 'f::fs-name

assumes $Trans$: $\Psi \triangleright P \mapsto R_s$
and FrP : $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $rAlpha$: $\bigwedge \Psi\ P\ A_P\ \Psi_P\ p\ R_s\ C. \llbracket A_P \#* \Psi; A_P \#* P; A_P \#* (p \cdot A_P); A_P \#* R_s; A_P \#* C; \rrbracket \implies$
 $set\ p \subseteq set\ A_P \times set(p \cdot A_P); distinctPerm\ p;$
 $Prop\ C\ \Psi\ P\ R_s\ A_P\ \Psi_P \rrbracket \implies Prop\ C\ \Psi\ P\ R_s\ (p$
 $\cdot A_P)\ (p \cdot \Psi_P)$
and $rInput$: $\bigwedge \Psi\ M\ K\ xvec\ N\ Tvec\ P\ C.$
 $\llbracket \Psi \vdash M \leftrightarrow K; distinct\ xvec; set\ xvec \subseteq supp\ N;$
 $length\ xvec = length\ Tvec; xvec \#* \Psi;$
 $xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (M(\lambda * xvec\ N).P)$
 $(K(\llbracket N[xvec::=Tvec] \rrbracket)) \prec (P[xvec::=Tvec])\ (\Box)\ (1)$
and $rBrInput$: $\bigwedge \Psi\ M\ K\ xvec\ N\ Tvec\ P\ C.$
 $\llbracket \Psi \vdash K \succeq M; distinct\ xvec; set\ xvec \subseteq supp\ N;$
 $length\ xvec = length\ Tvec; xvec \#* \Psi;$
 $xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (M(\lambda * xvec\ N).P)$
 $(\downarrow K(\llbracket N[xvec::=Tvec] \rrbracket)) \prec (P[xvec::=Tvec])\ (\Box)\ (1)$
and $rOutput$: $\bigwedge \Psi\ M\ K\ N\ P\ C. \Psi \vdash M \leftrightarrow K \implies Prop\ C\ \Psi\ (M\langle N \rangle.P)\ (K\langle N \rangle$
 $\prec P)\ (\Box)\ (1)$
and $rBrOutput$: $\bigwedge \Psi\ M\ K\ N\ P\ C. \Psi \vdash M \preceq K \implies Prop\ C\ \Psi\ (M\langle N \rangle.P)$
 $(\downarrow K\langle N \rangle \prec P)\ (\Box)\ (1)$
and $rCase$: $\bigwedge \Psi\ P\ R_s\ \varphi\ C_s\ A_P\ \Psi_P\ C. \llbracket \Psi \triangleright P \mapsto R_s; extractFrame\ P = \langle A_P,$
 $\Psi_P \rangle; distinct\ A_P; \bigwedge C. Prop\ C\ \Psi\ P\ R_s\ A_P\ \Psi_P;$
 $(\varphi, P) \in set\ C_s; \Psi \vdash \varphi; guarded\ P; \Psi_P \simeq \mathbf{1};$
 $(supp\ \Psi_P) = (\{\}::name\ set);$
 $A_P \#* \Psi; A_P \#* P; A_P \#* R_s; A_P \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (Cases\ C_s)\ R_s\ (\Box)\ (1)$
and $rPar1$: $\bigwedge \Psi\ \Psi_Q\ P\ \alpha\ P'\ A_Q\ Q\ A_P\ \Psi_P\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P';$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ (\alpha \prec P')\ A_P\ \Psi_P; distinct(bn\ \alpha);$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* \alpha; A_P \#* P'; A_P \#* A_Q; A_P$
 $\#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* \alpha; A_Q \#* P'; A_Q \#* \Psi_P;$
 $bn\ \alpha \#* \Psi; bn\ \alpha \#* P; bn\ \alpha \#* Q; bn\ \alpha \#* subject\ \alpha; bn\ \alpha \#* \Psi_P;$
 $bn\ \alpha \#* \Psi_Q;$
 $A_P \#* C; A_Q \#* C; bn\ \alpha \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (P \parallel Q)\ (\alpha \prec (P' \parallel Q))\ (A_P @ A_Q)\ (\Psi_P \otimes \Psi_Q)$

and $rPar2$: $\bigwedge \Psi \Psi_P Q \alpha Q' A_P P A_Q \Psi_Q C$.
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q' \rrbracket$;
 $extractFrame P = \langle A_P, \Psi_P \rangle$; $distinct A_P$;
 $extractFrame Q = \langle A_Q, \Psi_Q \rangle$; $distinct A_Q$;
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (\alpha \prec Q') A_Q \Psi_Q$; $distinct(bn \alpha)$;
 $A_P \#* P$; $A_P \#* Q$; $A_P \#* \Psi$; $A_P \#* \alpha$; $A_P \#* Q'$; $A_P \#* A_Q$; A_P
 $\#* \Psi_Q$;
 $A_Q \#* P$; $A_Q \#* Q$; $A_Q \#* \Psi$; $A_Q \#* \alpha$; $A_Q \#* Q'$; $A_Q \#* \Psi_P$;
 $bn \alpha \#* \Psi$; $bn \alpha \#* P$; $bn \alpha \#* Q$; $bn \alpha \#* subject \alpha$; $bn \alpha \#* \Psi_P$;
 $bn \alpha \#* \Psi_Q$;
 $A_P \#* C$; $A_Q \#* C$; $bn \alpha \#* C \rrbracket \implies$
 $Prop C \Psi (P \parallel Q) (\alpha \prec (P \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rComm1$: $\bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q K xvec Q' A_Q C$.
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(N) \prec P' \rrbracket$; $extractFrame P = \langle A_P, \Psi_P \rangle$;
 $distinct A_P$;
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P ((M(N)) \prec P') A_P \Psi_P$;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * xvec)\langle N \rangle \prec Q'$; $extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle$; $distinct A_Q$;
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K$;
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (K(\nu * xvec)\langle N \rangle \prec Q') A_Q \Psi_Q$; $distinct$
 $xvec$;
 $A_P \#* \Psi$; $A_P \#* \Psi_Q$; $A_P \#* P$; $A_P \#* M$; $A_P \#* N$; $A_P \#* P'$;
 $A_P \#* Q$; $A_P \#* Q'$; $A_P \#* A_Q$; $A_P \#* xvec$; $A_Q \#* \Psi$; $A_Q \#* \Psi_P$;
 $A_Q \#* P$; $A_Q \#* N$; $A_Q \#* P'$; $A_Q \#* Q$; $A_Q \#* K$; $A_Q \#* Q'$;
 $A_Q \#* xvec$; $xvec \#* \Psi$; $xvec \#* \Psi_P$; $xvec \#* \Psi_Q$; $xvec \#* P$; $xvec \#*$
 M ;
 $xvec \#* Q$; $xvec \#* K$; $A_P \#* C$; $A_Q \#* C$; $xvec \#* C \rrbracket \implies$
 $Prop C \Psi (P \parallel Q) (\tau \prec (\nu * xvec)(P' \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rComm2$: $\bigwedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q K Q' A_Q C$.
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P' \rrbracket$; $extractFrame P = \langle A_P,$
 $\Psi_P \rangle$; $distinct A_P$;
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P (M(\nu * xvec)\langle N \rangle \prec P') A_P \Psi_P$;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(N) \prec Q'$; $extractFrame Q = \langle A_Q, \Psi_Q \rangle$;
 $distinct A_Q$;
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (K(N) \prec Q') A_Q \Psi_Q$;
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K$; $distinct xvec$;
 $A_P \#* \Psi$; $A_P \#* \Psi_Q$; $A_P \#* P$; $A_P \#* M$; $A_P \#* N$; $A_P \#* P'$;
 $A_P \#* Q$; $A_P \#* Q'$; $A_P \#* A_Q$; $A_P \#* xvec$; $A_Q \#* \Psi$; $A_Q \#* \Psi_P$;
 $A_Q \#* P$; $A_Q \#* N$; $A_Q \#* P'$; $A_Q \#* Q$; $A_Q \#* K$; $A_Q \#* Q'$;
 $A_Q \#* xvec$; $xvec \#* \Psi$; $xvec \#* \Psi_P$; $xvec \#* \Psi_Q$; $xvec \#* P$; $xvec \#*$
 M ;
 $xvec \#* Q$; $xvec \#* K$; $A_P \#* C$; $A_Q \#* C$; $xvec \#* C \rrbracket \implies$
 $Prop C \Psi (P \parallel Q) (\tau \prec (\nu * xvec)(P' \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rBrMerge$: $\bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q C$.
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto iM(N) \prec P' \rrbracket$; $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P$
 $(iM(N) \prec P') A_P \Psi_P$;
 $extractFrame P = \langle A_P, \Psi_P \rangle$; $distinct A_P$;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto iM(N) \prec Q'$; $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q$
 $(iM(N) \prec Q') A_Q \Psi_Q$;

$extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M \implies$
 $Prop C \Psi (P \parallel Q) (iM(\downarrow N) \prec (P' \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rBrComm1: \wedge \Psi \Psi_Q P M N P' A_P \Psi_P Q xvec Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto iM(\downarrow N) \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
 $distinct A_P;$
 $\wedge C. Prop C (\Psi \otimes \Psi_Q) P ((iM(\downarrow N)) \prec P') A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto iM(\nu * xvec) \langle N \rangle \prec Q'; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle; distinct A_Q;$
 $\wedge C. Prop C (\Psi \otimes \Psi_P) Q (iM(\nu * xvec) \langle N \rangle \prec Q') A_Q \Psi_Q; distinct$
 $xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$
 $xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M \implies$
 $Prop C \Psi (P \parallel Q) (iM(\nu * xvec) \langle N \rangle \prec (P' \parallel Q')) (A_P @ A_Q) (\Psi_P$
 $\otimes \Psi_Q)$
and $rBrComm2: \wedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto iM(\nu * xvec) \langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle; distinct A_P;$
 $\wedge C. Prop C (\Psi \otimes \Psi_Q) P (iM(\nu * xvec) \langle N \rangle \prec P') A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto iM(\downarrow N) \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$
 $\wedge C. Prop C (\Psi \otimes \Psi_P) Q (iM(\downarrow N) \prec Q') A_Q \Psi_Q; distinct xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$
 $xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M \implies$
 $Prop C \Psi (P \parallel Q) (iM(\nu * xvec) \langle N \rangle \prec (P' \parallel Q')) (A_P @ A_Q) (\Psi_P$
 $\otimes \Psi_Q)$
and $rBrClose: \wedge \Psi P M xvec N P' A_P \Psi_P x C.$
 $\llbracket \Psi \triangleright P \mapsto iM(\nu * xvec) \langle N \rangle \prec P';$
 $x \in supp M;$
 $\wedge C. Prop C \Psi P (iM(\nu * xvec) \langle N \rangle \prec P') A_P \Psi_P;$
 $extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* xvec;$
 $distinct xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P;$
 $xvec \#* M;$
 $x \# \Psi; x \# xvec; x \# A_P;$
 $A_P \#* C; xvec \#* C; x \# C \implies$
 $Prop C \Psi ((\nu x) P) (\tau \prec ((\nu x) ((\nu * xvec) P))) (x \# A_P) \Psi_P$

and $rOpen$: $\bigwedge \Psi P M xvec yvec N P' x A_P \Psi_P C$.
 $\llbracket \Psi \triangleright P \mapsto M(\nu^*(xvec@yvec)) \langle N \rangle \prec P' ; extractFrame P = \langle A_P, \Psi_P \rangle ; distinct A_P ;$
 $\bigwedge C. Prop C \Psi P (M(\nu^*(xvec@yvec)) \langle N \rangle \prec P') A_P \Psi_P ; x \in supp$
 $N ; x \# \Psi ; x \# M ;$
 $x \# A_P ; x \# xvec ; x \# yvec ; A_P \#* \Psi ; A_P \#* P ; A_P \#* M ; A_P \#*$
 $N ; A_P \#* P' ;$
 $A_P \#* xvec ; A_P \#* yvec ; xvec \#* yvec ; distinct xvec ; distinct yvec ;$
 $xvec \#* \Psi ; xvec \#* P ; xvec \#* M ; xvec \#* \Psi_P ; yvec \#* \Psi_P ;$
 $yvec \#* \Psi ; yvec \#* P ; yvec \#* M ; A_P \#* C ; x \# C ; xvec \#* C ; yvec$
 $\#* C \rrbracket \implies$
 $Prop C \Psi ((\nu x)P) (M(\nu^*(xvec@x\#yvec)) \langle N \rangle \prec P') (x\#A_P) \Psi_P$
and $rBrOpen$: $\bigwedge \Psi P M xvec yvec N P' x A_P \Psi_P C$.
 $\llbracket \Psi \triangleright P \mapsto {}_iM(\nu^*(xvec@yvec)) \langle N \rangle \prec P' ; extractFrame P = \langle A_P, \Psi_P \rangle ; distinct A_P ;$
 $\bigwedge C. Prop C \Psi P ({}_iM(\nu^*(xvec@yvec)) \langle N \rangle \prec P') A_P \Psi_P ; x \in$
 $supp N ; x \# \Psi ; x \# M ;$
 $x \# A_P ; x \# xvec ; x \# yvec ; A_P \#* \Psi ; A_P \#* P ; A_P \#* M ; A_P \#*$
 $N ; A_P \#* P' ;$
 $A_P \#* xvec ; A_P \#* yvec ; xvec \#* yvec ; distinct xvec ; distinct yvec ;$
 $xvec \#* \Psi ; xvec \#* P ; xvec \#* M ; xvec \#* \Psi_P ; yvec \#* \Psi_P ;$
 $yvec \#* \Psi ; yvec \#* P ; yvec \#* M ; A_P \#* C ; x \# C ; xvec \#* C ; yvec$
 $\#* C \rrbracket \implies$
 $Prop C \Psi ((\nu x)P) ({}_iM(\nu^*(xvec@x\#yvec)) \langle N \rangle \prec P') (x\#A_P) \Psi_P$
and $rScope$: $\bigwedge \Psi P \alpha P' x A_P \Psi_P C$.
 $\llbracket \Psi \triangleright P \mapsto \alpha \prec P' ; extractFrame P = \langle A_P, \Psi_P \rangle ; distinct A_P ;$
 $\bigwedge C. Prop C \Psi P (\alpha \prec P') A_P \Psi_P ;$
 $x \# \Psi ; x \# \alpha ; x \# A_P ; A_P \#* \Psi ; A_P \#* P ;$
 $A_P \#* \alpha ; A_P \#* P' ; distinct(bn \alpha) ;$
 $bn \alpha \#* \Psi ; bn \alpha \#* P ; bn \alpha \#* subject \alpha ; bn \alpha \#* \Psi_P ;$
 $A_P \#* C ; x \# C ; bn \alpha \#* C \rrbracket \implies$
 $Prop C \Psi ((\nu x)P) (\alpha \prec ((\nu x)P')) (x\#A_P) \Psi_P$
and $rBang$: $\bigwedge \Psi P Rs A_P \Psi_P C$.
 $\llbracket \Psi \triangleright P \parallel !P \mapsto Rs ; guarded P ; extractFrame P = \langle A_P, \Psi_P \rangle ;$
 $distinct A_P ;$
 $\bigwedge C. Prop C \Psi (P \parallel !P) Rs A_P (\Psi_P \otimes \mathbf{1}) ; \Psi_P \simeq \mathbf{1} ; supp \Psi_P =$
 $(\{\} :: name set) ;$
 $A_P \#* \Psi ; A_P \#* P ; A_P \#* Rs ; A_P \#* C \rrbracket \implies Prop C \Psi (!P) Rs$
 $(\parallel) (1)$
shows $Prop C \Psi P Rs A_P \Psi_P$
 $\langle proof \rangle$

lemma $semanticsFrameInduct'$ [consumes 5, case-names $cAlpha$ $cFrameAlpha$ $cInput$ $cBrInput$ $cOutput$ $cBrOutput$ $cCase$ $cPar1$ $cPar2$ $cComm1$ $cComm2$ $cBrMerge$ $cBrComm1$ $cBrComm2$ $cBrClose$ $cOpen$ $cBrOpen$ $cScope$ $cBang$]:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) psi$
and Rs $:: ('a, 'b, 'c) residual$
and A_P $:: name list$

and $\Psi_P :: 'b$
and $Prop :: 'f::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow 'a \text{ action} \Rightarrow ('a, 'b, 'c) psi \Rightarrow name \text{ list} \Rightarrow 'b \Rightarrow bool$
and $C :: 'f::fs-name$

assumes $Trans: \Psi \triangleright P \mapsto \alpha \prec P'$
and $FrP: extractFrame P = \langle A_P, \Psi_P \rangle$
and $distinct A_P$
and $bn \alpha \#* subject \alpha$
and $distinct(bn \alpha)$
and $rAlpha: \bigwedge \Psi P \alpha P' p A_P \Psi_P C. \llbracket bn \alpha \#* \Psi; bn \alpha \#* P; bn \alpha \#* subject \alpha; bn \alpha \#* \Psi_P; A_P \#* \alpha; A_P \#* P'; A_P \#* C; bn \alpha \#* C; bn \alpha \#* (p \cdot \alpha); A_P \#* \Psi; A_P \#* P; set p \subseteq set(bn \alpha) \times set(bn(p \cdot \alpha)); distinctPerm p; bn(p \cdot \alpha) \#* \alpha; (bn(p \cdot \alpha)) \#* P'; Prop C \Psi P \alpha P' A_P \Psi_P \rrbracket \Longrightarrow Prop C \Psi P (p \cdot \alpha) (p \cdot P') A_P \Psi_P$
and $rFrameAlpha: \bigwedge \Psi P A_P \Psi_P p \alpha P' C. \llbracket A_P \#* \Psi; A_P \#* P; A_P \#* (p \cdot A_P); A_P \#* \alpha; A_P \#* P'; A_P \#* C; set p \subseteq set A_P \times set(p \cdot A_P); distinctPerm p; A_P \#* subject \alpha; Prop C \Psi P \alpha P' A_P \Psi_P \rrbracket \Longrightarrow Prop C \Psi P \alpha P' (p \cdot A_P) (p \cdot \Psi_P)$
and $rInput: \bigwedge \Psi M K xvec N Tvec P C. \llbracket \Psi \vdash M \leftrightarrow K; distinct xvec; set xvec \subseteq supp N; length xvec = length Tvec; xvec \#* \Psi; xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \Longrightarrow Prop C \Psi (M(\lambda * xvec N).P) (K(\llbracket N[xvec::=Tvec] \rrbracket)) (P[xvec::=Tvec]) (\llbracket \rrbracket) (\mathbf{1})$
and $rBrInput: \bigwedge \Psi M K xvec N Tvec P C. \llbracket \Psi \vdash K \succeq M; distinct xvec; set xvec \subseteq supp N; length xvec = length Tvec; xvec \#* \Psi; xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \Longrightarrow Prop C \Psi (M(\lambda * xvec N).P) (\llbracket K(\llbracket N[xvec::=Tvec] \rrbracket) \rrbracket) (P[xvec::=Tvec]) (\llbracket \rrbracket) (\mathbf{1})$
and $rOutput: \bigwedge \Psi M K N P C. \Psi \vdash M \leftrightarrow K \Longrightarrow Prop C \Psi (M\langle N \rangle.P) (K\langle N \rangle) P (\llbracket \rrbracket) (\mathbf{1})$
and $rBrOutput: \bigwedge \Psi M K N P C. \Psi \vdash M \preceq K \Longrightarrow Prop C \Psi (M\langle N \rangle.P) (\llbracket K\langle N \rangle \rrbracket) P (\llbracket \rrbracket) (\mathbf{1})$
and $rCase: \bigwedge \Psi P \alpha P' \varphi Cs A_P \Psi_P C. \llbracket \Psi \triangleright P \mapsto \alpha \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P; \bigwedge C. Prop C \Psi P \alpha P' A_P \Psi_P; (\varphi, P) \in set Cs; \Psi \vdash \varphi; guarded P; \Psi_P \simeq \mathbf{1}; (supp \Psi_P) = (\{ \}::name \text{ set}); A_P \#* \Psi; A_P \#* P; A_P \#* \alpha; A_P \#* P'; A_P \#* C \rrbracket \Longrightarrow Prop C \Psi (Cases Cs) \alpha P' (\llbracket \rrbracket) (\mathbf{1})$
and $rPar1: \bigwedge \Psi \Psi_Q P \alpha P' A_Q Q A_P \Psi_P C. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P';$

$extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P \alpha P' A_P \Psi_P;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* \alpha; A_P \#* P'; A_P \#* A_Q; A_P$
 $\#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* \alpha; A_Q \#* P'; A_Q \#* \Psi_P;$
 $bn \alpha \#* \Psi; bn \alpha \#* P; bn \alpha \#* Q; bn \alpha \#* subject \alpha; bn \alpha \#* \Psi_P;$
 $bn \alpha \#* \Psi_Q;$
 $A_P \#* C; A_Q \#* C; bn \alpha \#* C \implies$
 $Prop C \Psi (P \parallel Q) \alpha (P' \parallel Q) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rPar2: \bigwedge \Psi \Psi_P Q \alpha Q' A_P P A_Q \Psi_Q C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q' \rrbracket;$
 $extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q \alpha Q' A_Q \Psi_Q;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* \alpha; A_P \#* Q'; A_P \#* A_Q; A_P$
 $\#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* \alpha; A_Q \#* Q'; A_Q \#* \Psi_P;$
 $bn \alpha \#* \Psi; bn \alpha \#* P; bn \alpha \#* Q; bn \alpha \#* subject \alpha; bn \alpha \#* \Psi_P;$
 $bn \alpha \#* \Psi_Q;$
 $A_P \#* C; A_Q \#* C; bn \alpha \#* C \implies$
 $Prop C \Psi (P \parallel Q) \alpha (P \parallel Q') (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rComm1: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q K xvec Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(N) \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
 $distinct A_P;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P (M(N)) P' A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * xvec)\langle N \rangle \prec Q'; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle; distinct A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; distinct xvec;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (K(\nu * xvec)\langle N \rangle) Q' A_Q \Psi_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#*$
 $M;$
 $xvec \#* Q; xvec \#* K; A_P \#* C; A_Q \#* C; xvec \#* C \implies$
 $Prop C \Psi (P \parallel Q) (\tau) (\nu * xvec)(P' \parallel Q') (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rComm2: \bigwedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q K Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle; distinct A_P;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P (M(\nu * xvec)\langle N \rangle) P' A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(N) \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (K(N)) Q' A_Q \Psi_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; distinct xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* K; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#*$

M ;

$xvec \#* Q; xvec \#* K; A_P \#* C; A_Q \#* C; xvec \#* C \implies$
 $Prop C \Psi (P \parallel Q) (\tau) (\nu * xvec) (P' \parallel Q') (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rBrMerge: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto iM(N) \prec P'; \bigwedge C. Prop C (\Psi \otimes \Psi_Q) P$
 $(iM(N)) P' A_P \Psi_P;$

$extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto iM(N) \prec Q'; \bigwedge C. Prop C (\Psi \otimes \Psi_P) Q$
 $(iM(N)) Q' A_Q \Psi_Q;$

$extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C \implies$

$Prop C \Psi (P \parallel Q) (iM(N)) (P' \parallel Q') (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rBrComm1: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q xvec Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto iM(N) \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
 $distinct A_P;$

$\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P (iM(N)) P' A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto iM(\nu * xvec) \langle N \rangle \prec Q'; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle; distinct A_Q;$

$distinct xvec;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (iM(\nu * xvec) \langle N \rangle) Q' A_Q \Psi_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$
 $xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M \implies$
 $Prop C \Psi (P \parallel Q) (iM(\nu * xvec) \langle N \rangle) (P' \parallel Q') (A_P @ A_Q) (\Psi_P \otimes$
 $\Psi_Q)$

and $rBrComm2: \bigwedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto iM(\nu * xvec) \langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle; distinct A_P;$

$\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P (iM(\nu * xvec) \langle N \rangle) P' A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto iM(N) \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$

$\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q (iM(N)) Q' A_Q \Psi_Q;$
 $distinct xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$
 $xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M \implies$
 $Prop C \Psi (P \parallel Q) (iM(\nu * xvec) \langle N \rangle) (P' \parallel Q') (A_P @ A_Q) (\Psi_P \otimes$
 $\Psi_Q)$

and $rBrClose: \bigwedge \Psi P M xvec N P' A_P \Psi_P x C.$
 $\llbracket \Psi \triangleright P \mapsto iM(\nu * xvec) \langle N \rangle \prec P';$

$x \in \text{supp } M$;
 $\bigwedge C. \text{Prop } C \Psi P (\text{i}M(\nu * \text{vec})\langle N \rangle) P' A_P \Psi_P$;
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle$; $\text{distinct } A_P$;
 $A_P \#* \Psi$; $A_P \#* P$; $A_P \#* M$; $A_P \#* N$; $A_P \#* P'$; $A_P \#* \text{vec}$;
 $\text{distinct } \text{vec}$; $\text{vec} \#* \Psi$; $\text{vec} \#* \Psi_P$; $\text{vec} \#* P$;
 $\text{vec} \#* M$;
 $x \# \Psi$; $x \# \text{vec}$; $x \# A_P$;
 $A_P \#* C$; $\text{vec} \#* C$; $x \# C \implies$
 $\text{Prop } C \Psi ((\nu x)P) (\tau) ((\nu x)(\nu * \text{vec})P')$ $(x \# A_P) \Psi_P$
and $rOpen$: $\bigwedge \Psi P M \text{vec } yvec N P' x A_P \Psi_P y C$.
 $\llbracket \Psi \triangleright P \mapsto M(\nu * (\text{vec} @ yvec))\langle N \rangle \prec P' \text{; extractFrame } P = \langle A_P,$
 $\Psi_P \rangle$; $\text{distinct } A_P$;
 $\bigwedge C. \text{Prop } C \Psi P (M(\nu * (\text{vec} @ yvec))\langle N \rangle) P' A_P \Psi_P$; $x \in \text{supp}$
 N ; $x \# \Psi$; $x \# M$;
 $x \# A_P$; $x \# \text{vec}$; $x \# yvec$; $A_P \#* \Psi$; $A_P \#* P$; $A_P \#* M$; $A_P \#*$
 N ; $A_P \#* P'$;
 $A_P \#* \text{vec}$; $A_P \#* yvec$; $\text{vec} \#* yvec$; $\text{distinct } \text{vec}$; $\text{distinct } yvec$;
 $\text{vec} \#* \Psi$; $\text{vec} \#* P$; $\text{vec} \#* M$; $\text{vec} \#* \Psi_P$;
 $yvec \#* \Psi$; $yvec \#* P$; $yvec \#* M$; $A_P \#* C$; $x \# C$; $\text{vec} \#* C$; $yvec$
 $\#* C$;
 $y \neq x$; $y \# \Psi$; $y \# P$; $y \# M$; $y \# \text{vec}$; $y \# yvec$; $y \# N$; $y \# P'$; $y \#$
 A_P ; $y \# \Psi_P$; $y \# C \implies$
 $\text{Prop } C \Psi ((\nu x)P) (M(\nu * (\text{vec} @ y \# yvec))\langle [(x, y)] \cdot N \rangle) ([(x,$
 $y)] \cdot P')$ $(x \# A_P) \Psi_P$
and $rBrOpen$: $\bigwedge \Psi P M \text{vec } yvec N P' x A_P \Psi_P y C$.
 $\llbracket \Psi \triangleright P \mapsto \text{i}M(\nu * (\text{vec} @ yvec))\langle N \rangle \prec P' \text{; extractFrame } P = \langle A_P,$
 $\Psi_P \rangle$; $\text{distinct } A_P$;
 $\bigwedge C. \text{Prop } C \Psi P (\text{i}M(\nu * (\text{vec} @ yvec))\langle N \rangle) P' A_P \Psi_P$; $x \in \text{supp}$
 N ; $x \# \Psi$; $x \# M$;
 $x \# A_P$; $x \# \text{vec}$; $x \# yvec$; $A_P \#* \Psi$; $A_P \#* P$; $A_P \#* M$; $A_P \#*$
 N ; $A_P \#* P'$;
 $A_P \#* \text{vec}$; $A_P \#* yvec$; $\text{vec} \#* yvec$; $\text{distinct } \text{vec}$; $\text{distinct } yvec$;
 $\text{vec} \#* \Psi$; $\text{vec} \#* P$; $\text{vec} \#* M$; $\text{vec} \#* \Psi_P$;
 $yvec \#* \Psi$; $yvec \#* P$; $yvec \#* M$; $A_P \#* C$; $x \# C$; $\text{vec} \#* C$; $yvec$
 $\#* C$;
 $y \neq x$; $y \# \Psi$; $y \# P$; $y \# M$; $y \# \text{vec}$; $y \# yvec$; $y \# N$; $y \# P'$; $y \#$
 A_P ; $y \# \Psi_P$; $y \# C \implies$
 $\text{Prop } C \Psi ((\nu x)P) (\text{i}M(\nu * (\text{vec} @ y \# yvec))\langle [(x, y)] \cdot N \rangle) ([(x,$
 $y)] \cdot P')$ $(x \# A_P) \Psi_P$
and $rScope$: $\bigwedge \Psi P \alpha P' x A_P \Psi_P C$.
 $\llbracket \Psi \triangleright P \mapsto \alpha \prec P' \text{; extractFrame } P = \langle A_P, \Psi_P \rangle$; $\text{distinct } A_P$;
 $\bigwedge C. \text{Prop } C \Psi P \alpha P' A_P \Psi_P$;
 $x \# \Psi$; $x \# \alpha$; $x \# A_P$; $A_P \#* \Psi$; $A_P \#* P$;
 $A_P \#* \alpha$; $A_P \#* P'$;
 $bn \alpha \#* \Psi$; $bn \alpha \#* P$; $bn \alpha \#* \text{subject } \alpha$; $bn \alpha \#* \Psi_P$;
 $A_P \#* C$; $x \# C$; $bn \alpha \#* C \implies$
 $\text{Prop } C \Psi ((\nu x)P) \alpha ((\nu x)P')$ $(x \# A_P) \Psi_P$
and $rBang$: $\bigwedge \Psi P \alpha P' A_P \Psi_P C$.
 $\llbracket \Psi \triangleright P \parallel !P \mapsto \alpha \prec P' \text{; guarded } P \text{; extractFrame } P = \langle A_P, \Psi_P \rangle$;

$distinct\ A_P;$
 $\bigwedge C. Prop\ C\ \Psi\ (P\ \parallel\ !P)\ \alpha\ P'\ A_P\ (\Psi_P\ \otimes\ \mathbf{1});\ \Psi_P\ \simeq\ \mathbf{1};\ supp\ \Psi_P$
 $=\ (\{\}\ :: name\ set);$
 $A_P\ \#\#\ \Psi;\ A_P\ \#\#\ P;\ A_P\ \#\#\ \alpha;\ A_P\ \#\#\ P';\ A_P\ \#\#\ C\ \Longrightarrow\ Prop\ C\ \Psi$
 $(!P)\ \alpha\ P'\ (\parallel)\ (\mathbf{1})$
shows $Prop\ C\ \Psi\ P\ \alpha\ P'\ A_P\ \Psi_P$
 $\langle proof \rangle$

lemma $inputFrameInduct[consumes\ 3,\ case-names\ cAlpha\ cInput\ cCase\ cPar1$
 $cPar2\ cScope\ cBang]:$

fixes $\Psi\ \ ::\ 'b$
and $P\ \ ::\ ('a,\ 'b,\ 'c)\ psi$
and $M\ \ ::\ 'a$
and $N\ \ ::\ 'a$
and $P'\ \ ::\ ('a,\ 'b,\ 'c)\ psi$
and $Prop\ \ ::\ 'f::fs-name\ \Rightarrow\ 'b\ \Rightarrow\ ('a,\ 'b,\ 'c)\ psi\ \Rightarrow$
 $\ 'a\ \Rightarrow\ 'a\ \Rightarrow\ ('a,\ 'b,\ 'c)\ psi\ \Rightarrow\ name\ list\ \Rightarrow\ 'b\ \Rightarrow\ bool$
and $C\ \ ::\ 'f::fs-name$

assumes $Trans: \Psi \triangleright P \mapsto M(\downarrow N) \prec P'$
and $FrP: extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $rAlpha: \bigwedge \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P\ p\ C. \llbracket A_P\ \#\#\ \Psi;\ A_P\ \#\#\ P;\ A_P\ \#\#\ M;\ A_P\ \#\#\$
 $N;\ A_P\ \#\#\ P';\ A_P\ \#\#\ (p \cdot A_P);\ A_P\ \#\#\ C \rrbracket \Longrightarrow$
 $set\ p \subseteq set\ A_P \times set(p \cdot A_P); distinctPerm\ p;$
 $Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P \rrbracket \Longrightarrow Prop\ C\ \Psi$

$P\ M\ N\ P'\ (p \cdot A_P)\ (p \cdot \Psi_P)$
and $rInput: \bigwedge \Psi\ M\ K\ xvec\ N\ Tvec\ P\ C.$
 $\llbracket \Psi \vdash M \leftrightarrow K; distinct\ xvec; set\ xvec \subseteq supp\ N;$
 $length\ xvec = length\ Tvec; xvec\ \#\#\ \Psi;$
 $xvec\ \#\#\ M; xvec\ \#\#\ K; xvec\ \#\#\ C \rrbracket \Longrightarrow$
 $Prop\ C\ \Psi\ (M(\lambda * xvec\ N).P)$
 $K\ (N[xvec ::= Tvec])\ (P[xvec ::= Tvec])\ (\parallel)\ (\mathbf{1})$
and $rCase: \bigwedge \Psi\ P\ M\ N\ P'\ \varphi\ Cs\ A_P\ \Psi_P\ C. \llbracket \Psi \triangleright P \mapsto M(\downarrow N) \prec P'; extractFrame$
 $P = \langle A_P, \Psi_P \rangle; distinct\ A_P; \bigwedge C. Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P;$
 $(\varphi, P) \in set\ Cs; \Psi \vdash \varphi; guarded\ P; \Psi_P \simeq \mathbf{1};$
 $(supp\ \Psi_P) = (\{\}\ :: name\ set);$

$A_P\ \#\#\ \Psi;\ A_P\ \#\#\ P;\ A_P\ \#\#\ M;\ A_P\ \#\#\ N;\ A_P$
 $\#\#\ P';\ A_P\ \#\#\ C \rrbracket \Longrightarrow Prop\ C\ \Psi\ (Cases\ Cs)\ M\ N\ P'\ (\parallel)\ (\mathbf{1})$

and $rPar1: \bigwedge \Psi\ \Psi_Q\ P\ M\ N\ P'\ A_Q\ Q\ A_P\ \Psi_P\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\downarrow N) \prec P';$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ M\ N\ P'\ A_P\ \Psi_P;$
 $A_P\ \#\#\ P;\ A_P\ \#\#\ Q;\ A_P\ \#\#\ \Psi;\ A_P\ \#\#\ M;\ A_P\ \#\#\ N;\ A_P\ \#\#\ P';\ A_P\ \#\#\$
 $A_Q;\ A_P\ \#\#\ \Psi_Q;$
 $A_Q\ \#\#\ P;\ A_Q\ \#\#\ Q;\ A_Q\ \#\#\ \Psi;\ A_Q\ \#\#\ M;\ A_Q\ \#\#\ N;\ A_Q\ \#\#\ P';\ A_Q$
 $\#\#\ \Psi_P;$
 $A_P\ \#\#\ C;\ A_Q\ \#\#\ C \rrbracket \Longrightarrow$

$Prop\ C\ \Psi\ (P\ \parallel\ Q)\ M\ N\ (P'\ \parallel\ Q)\ (A_P @ A_Q)\ (\Psi_P \otimes \Psi_Q)$
and $rPar2: \bigwedge \Psi\ \Psi_P\ Q\ M\ N\ Q'\ A_P\ P\ A_Q\ \Psi_Q\ C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto M(N) \prec Q' \rrbracket;$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ M\ N\ Q'\ A_Q\ \Psi_Q;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* N; A_P \#* Q'; A_P$
 $\#* A_Q; A_P \#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* N; A_Q \#* Q'; A_Q$
 $\#* \Psi_P;$
 $A_P \#* C; A_Q \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (P\ \parallel\ Q)\ M\ N\ (P\ \parallel\ Q')\ (A_P @ A_Q)\ (\Psi_P \otimes \Psi_Q)$
and $rScope: \bigwedge \Psi\ P\ M\ N\ P'\ x\ A_P\ \Psi_P\ C.$
 $\llbracket \Psi \triangleright P \mapsto M(N) \prec P'; extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $\bigwedge C. Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P; x \# \Psi; x \# M; x \# N;$
 $x \# A_P; A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$
 $A_P \#* C; x \# C \rrbracket \implies$
 $Prop\ C\ \Psi\ ((\nu x)P)\ M\ N\ ((\nu x)P')\ (x \# A_P)\ \Psi_P$
and $rBang: \bigwedge \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P\ C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto M(N) \prec P'; guarded\ P; extractFrame\ P = \langle A_P,$
 $\Psi_P \rangle; distinct\ A_P;$
 $\bigwedge C. Prop\ C\ \Psi\ (P \parallel !P)\ M\ N\ P'\ A_P\ (\Psi_P \otimes \mathbf{1}); \Psi_P \simeq \mathbf{1}; (supp$
 $\Psi_P) = (\{\} :: name\ set);$
 $A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* C \rrbracket \implies$
 $Prop\ C\ \Psi\ (!P)\ M\ N\ P'\ (\mathbf{1})\ (\mathbf{1})$
shows $Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P$
 $\langle proof \rangle$

lemma $brinputFrameInduct[consumes\ 3, case-names\ cAlpha\ cBrInput\ cCase\ cPar1\ cPar2\ cBrMerge\ cScope\ cBang]:$

fixes $\Psi \quad :: 'b$
and $P \quad :: ('a, 'b, 'c)\ psi$
and $M \quad :: 'a$
and $N \quad :: 'a$
and $P' \quad :: ('a, 'b, 'c)\ psi$
and $Prop \quad :: 'f :: fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c)\ psi \Rightarrow$
 $'a \Rightarrow 'a \Rightarrow ('a, 'b, 'c)\ psi \Rightarrow name\ list \Rightarrow 'b \Rightarrow bool$
and $C \quad :: 'f :: fs-name$

assumes $Trans: \Psi \triangleright P \mapsto_i M(N) \prec P'$

and $FrP: extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $rAlpha: \bigwedge \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P\ p\ C. \llbracket A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#*$
 $N; A_P \#* P'; A_P \#* (p \cdot A_P); A_P \#* C;$
 $set\ p \subseteq set\ A_P \times set(p \cdot A_P); distinctPerm\ p;$
 $Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P \rrbracket \implies Prop\ C\ \Psi$
 $P\ M\ N\ P'\ (p \cdot A_P)\ (p \cdot \Psi_P)$
and $rBrInput: \bigwedge \Psi\ M\ K\ xvec\ N\ Tvec\ P\ C.$
 $\llbracket \Psi \vdash K \succeq M; distinct\ xvec; set\ xvec \subseteq supp\ N;$

$length\ xvec = length\ Tvec; xvec \#* \Psi;$
 $xvec \#* M; xvec \#* K; xvec \#* C \Longrightarrow$
 $Prop\ C\ \Psi\ (M(\lambda*xvec\ N).P)$
 $K\ (N[xvec::=Tvec])\ (P[xvec::=Tvec])\ (\Box)\ (\mathbf{1})$

and $rCase: \bigwedge \Psi\ P\ M\ N\ P'\ \varphi\ Cs\ A_P\ \Psi_P\ C. \llbracket \Psi \triangleright P \mapsto_i M(N) \prec P' \rrbracket;$ *extract-Frame* $P = \langle A_P, \Psi_P \rangle;$ *distinct* $A_P;$ $\bigwedge C. Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P;$
 $(\varphi, P) \in set\ Cs; \Psi \vdash \varphi; guarded\ P; \Psi_P \simeq \mathbf{1};$
 $(supp\ \Psi_P) = (\{\}::name\ set);$

$A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* N; A_P$
 $\#* P'; A_P \#* C \Longrightarrow Prop\ C\ \Psi\ (Cases\ Cs)\ M\ N\ P'\ (\Box)\ (\mathbf{1})$

and $rPar1: \bigwedge \Psi\ \Psi_Q\ P\ M\ N\ P'\ A_Q\ Q\ A_P\ \Psi_P\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P' \rrbracket;$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ M\ N\ P'\ A_P\ \Psi_P;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#*$
 $A_Q; A_P \#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* N; A_Q \#* P'; A_Q$
 $\#* \Psi_P;$
 $A_P \#* C; A_Q \#* C \Longrightarrow$
 $Prop\ C\ \Psi\ (P \parallel Q)\ M\ N\ (P' \parallel Q)\ (A_P @ A_Q)\ (\Psi_P \otimes \Psi_Q)$

and $rPar2: \bigwedge \Psi\ \Psi_P\ Q\ M\ N\ Q'\ A_P\ P\ A_Q\ \Psi_Q\ C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M(N) \prec Q' \rrbracket;$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $\bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ M\ N\ Q'\ A_Q\ \Psi_Q;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* N; A_P \#* Q'; A_P$
 $\#* A_Q; A_P \#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* N; A_Q \#* Q'; A_Q$
 $\#* \Psi_P;$
 $A_P \#* C; A_Q \#* C \Longrightarrow$
 $Prop\ C\ \Psi\ (P \parallel Q)\ M\ N\ (P' \parallel Q')\ (A_P @ A_Q)\ (\Psi_P \otimes \Psi_Q)$

and $rBrMerge: \bigwedge \Psi\ \Psi_Q\ P\ M\ N\ P'\ A_P\ \Psi_P\ Q\ Q'\ A_Q\ C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P' \rrbracket; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_Q)\ P\ M\ N$
 $P'\ A_P\ \Psi_P;$
 $extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(N) \prec Q'; \bigwedge C. Prop\ C\ (\Psi \otimes \Psi_P)\ Q\ M\ N$
 $Q'\ A_Q\ \Psi_Q;$
 $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C \Longrightarrow$
 $Prop\ C\ \Psi\ (P \parallel Q)\ M\ N\ (P' \parallel Q')\ (A_P @ A_Q)\ (\Psi_P \otimes \Psi_Q)$

and $rScope: \bigwedge \Psi\ P\ M\ N\ P'\ x\ A_P\ \Psi_P\ C.$
 $\llbracket \Psi \triangleright P \mapsto_i M(N) \prec P' \rrbracket; extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct$
 $A_P;$
 $\bigwedge C. Prop\ C\ \Psi\ P\ M\ N\ P'\ A_P\ \Psi_P; x \# \Psi; x \# M; x \# N;$
 $x \# A_P; A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P';$

$A_P \#* C; x \# C \implies$
 $Prop C \Psi (\nu x)P M N (\nu x)P' (x\#A_P) \Psi_P$
and *rBang*: $\bigwedge \Psi P M N P' A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto_i M(N) \prec P'; \text{ guarded } P; \text{ extractFrame } P =$
 $\langle A_P, \Psi_P \rangle; \text{ distinct } A_P;$
 $\bigwedge C. Prop C \Psi (P \parallel !P) M N P' A_P (\Psi_P \otimes \mathbf{1}); \Psi_P \simeq \mathbf{1}; (\text{supp}$
 $\Psi_P) = (\{\}::\text{name set});$
 $A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* C \implies$
 $Prop C \Psi (!P) M N P' (\mathbf{1}) (\mathbf{1})$
shows $Prop C \Psi P M N P' A_P \Psi_P$
<proof>

lemma *outputFrameInduct*[*consumes 3, case-names cAlpha cOutput cCase cPar1 cPar2 cOpen cScope cBang*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $B :: ('a, 'b, 'c) boundOutput$
and $A_P :: \text{name list}$
and $\Psi_P :: 'b$
and $Prop :: 'f::fs\text{-name} \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow$
 $'a \Rightarrow ('a, 'b, 'c) boundOutput \Rightarrow \text{name list} \Rightarrow 'b \Rightarrow \text{bool}$
and $C :: 'f::fs\text{-name}$

assumes *Trans*: $\Psi \triangleright P \mapsto ROut M B$

and *FrP*: $\text{extractFrame } P = \langle A_P, \Psi_P \rangle$
and *distinct* A_P
and *rAlpha*: $\bigwedge \Psi P M A_P \Psi_P p B C. \llbracket A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* (p$
 $\cdot A_P); A_P \#* B; A_P \#* C;$
 $\text{set } p \subseteq \text{set } A_P \times \text{set}(p \cdot A_P); \text{ distinctPerm } p;$
 $Prop C \Psi P M B A_P \Psi_P \implies Prop C \Psi P M B$
 $(p \cdot A_P) (p \cdot \Psi_P)$
and *rOutput*: $\bigwedge \Psi M K N P C. \Psi \vdash M \leftrightarrow K \implies Prop C \Psi (M\langle N \rangle.P) K (N$
 $\prec' P) (\mathbf{1}) (\mathbf{1})$
and *rCase*: $\bigwedge \Psi P M B \varphi Cs A_P \Psi_P C. \llbracket \Psi \triangleright P \mapsto (ROut M B); \text{ extractFrame}$
 $P = \langle A_P, \Psi_P \rangle; \text{ distinct } A_P; \bigwedge C. Prop C \Psi P M B A_P \Psi_P;$
 $(\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{ guarded } P; \Psi_P \simeq \mathbf{1};$
 $(\text{supp } \Psi_P) = (\{\}::\text{name set});$

$A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* B; A_P \#*$
 $C \implies Prop C \Psi (Cases Cs) M B (\mathbf{1}) (\mathbf{1})$

and *rPar1*: $\bigwedge \Psi \Psi_Q P M \text{vec } N P' A_Q Q A_P \Psi_P C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * \text{vec})\langle N \rangle \prec P';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{ distinct } A_P;$
 $\text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{ distinct } A_Q;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P M (\nu * \text{vec})N \prec' P' A_P \Psi_P;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* \text{vec}; A_P \#* N; A_P$
 $\#* P'; A_P \#* A_Q; A_P \#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* \text{vec}; A_Q \#* N; A_Q$
 $\#* P'; A_Q \#* \Psi_P;$

$xvec \#* \Psi; xvec \#* P; xvec \#* Q; xvec \#* M; xvec \#* \Psi_P; xvec \#* \Psi_Q;$
 $A_P \#* C; A_Q \#* C; xvec \#* C \implies$
 $Prop C \Psi (P \parallel Q) M ((\nu*xvec)N \prec' (P' \parallel Q)) (A_P @ A_Q) (\Psi_P \otimes$
 $\Psi_Q)$
and $rPar2: \bigwedge \Psi \Psi_P Q M xvec N Q' A_P P A_Q \Psi_Q C.$
 $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto M(\nu*xvec)\langle N \rangle \prec Q' ;$
 $extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q M ((\nu*xvec)N \prec' Q') A_Q \Psi_Q;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* xvec; A_P \#* N; A_P$
 $\#* Q'; A_P \#* A_Q; A_P \#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* xvec; A_Q \#* N; A_Q$
 $\#* Q'; A_Q \#* \Psi_P;$
 $xvec \#* \Psi; xvec \#* P; xvec \#* Q; xvec \#* M; xvec \#* \Psi_P; xvec \#* \Psi_Q;$
 $A_P \#* C; A_Q \#* C; xvec \#* C \implies$
 $Prop C \Psi (P \parallel Q) M ((\nu*xvec)N \prec' (P \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes$
 $\Psi_Q)$
and $rOpen: \bigwedge \Psi P M xvec yvec N P' x A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \mapsto M(\nu*(xvec@yvec))\langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle; distinct A_P;$
 $\bigwedge C. Prop C \Psi P M ((\nu*(xvec@yvec))N \prec' P') A_P \Psi_P; x \in supp$
 $N; x \# \Psi; x \# M;$
 $x \# A_P; x \# xvec; x \# yvec; A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#*$
 $N; A_P \#* P';$
 $A_P \#* xvec; A_P \#* yvec;$
 $xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* \Psi_P;$
 $yvec \#* \Psi; yvec \#* P; yvec \#* M; A_P \#* C; x \# C; xvec \#* C; yvec$
 $\#* C \implies$
 $Prop C \Psi ((\nu x)P) M ((\nu*(xvec@x\#yvec))N \prec' P') (x\#A_P) \Psi_P$
and $rScope: \bigwedge \Psi P M xvec N P' x A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
 $distinct A_P;$
 $\bigwedge C. Prop C \Psi P M ((\nu*xvec)N \prec' P') A_P \Psi_P;$
 $x \# \Psi; x \# M; x \# xvec; x \# N; x \# A_P; A_P \#* \Psi; A_P \#* P;$
 $A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* xvec;$
 $xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* \Psi_P;$
 $A_P \#* C; x \# C; xvec \#* C \implies$
 $Prop C \Psi ((\nu x)P) M ((\nu*xvec)N \prec' ((\nu x)P')) (x\#A_P) \Psi_P$
and $rBang: \bigwedge \Psi P M B A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto ROut M B; guarded P; extractFrame P = \langle A_P,$
 $\Psi_P \rangle; distinct A_P;$
 $\bigwedge C. Prop C \Psi (P \parallel !P) M B A_P (\Psi_P \otimes \mathbf{1}); \Psi_P \simeq \mathbf{1}; supp \Psi_P$
 $= (\{\}::name set);$
 $A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* C \implies Prop C \Psi (!P) M$
 $B (\parallel) (\mathbf{1})$
shows $Prop C \Psi P M B A_P \Psi_P$
 $\langle proof \rangle$

lemma $brouputFrameInduct[consumes 3, case-names cAlpha cBrOutput cCase$

cPar1 cPar2 cBrComm1 cBrComm2 cBrOpen cScope cBang]:

```

fixes  $\Psi$   :: 'b
and  $P$     :: ('a, 'b, 'c) psi
and  $M$     :: 'a
and  $B$     :: ('a, 'b, 'c) boundOutput
and  $A_P$   :: name list
and  $\Psi_P$  :: 'b
and  $Prop$  :: 'f::fs-name  $\Rightarrow$  'b  $\Rightarrow$  ('a, 'b, 'c) psi  $\Rightarrow$ 
           'a  $\Rightarrow$  ('a, 'b, 'c) boundOutput  $\Rightarrow$  name list  $\Rightarrow$  'b  $\Rightarrow$  bool
and  $C$     :: 'f::fs-name

assumes Trans:  $\Psi \triangleright P \mapsto RBrOut M B$ 
and FrP: extractFrame  $P = \langle A_P, \Psi_P \rangle$ 
and distinct  $A_P$ 
and rAlpha:  $\bigwedge \Psi P M A_P \Psi_P p B C. \llbracket A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* (p \cdot A_P); A_P \#* B; A_P \#* C; \rrbracket$ 
            $set p \subseteq set A_P \times set(p \cdot A_P); distinctPerm p;$ 
            $Prop C \Psi P M B A_P \Psi_P \implies Prop C \Psi P M B$ 
            $(p \cdot A_P) (p \cdot \Psi_P)$ 
and rBrOutput:  $\bigwedge \Psi M K N P C. \Psi \vdash M \preceq K \implies Prop C \Psi (M \langle N \rangle . P) K$ 
            $(N \prec' P) (\llbracket \rrbracket) (1)$ 
and rCase:  $\bigwedge \Psi P M B \varphi Cs A_P \Psi_P C. \llbracket \Psi \triangleright P \mapsto (RBrOut M B); extractFrame$ 
            $P = \langle A_P, \Psi_P \rangle; distinct A_P; \bigwedge C. Prop C \Psi P M B A_P \Psi_P;$ 
            $(\varphi, P) \in set Cs; \Psi \vdash \varphi; guarded P; \Psi_P \simeq \mathbf{1};$ 
            $(supp \Psi_P) = (\{\}::name set);$ 
            $A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* B; A_P \#*$ 
            $C \rrbracket \implies Prop C \Psi (Cases Cs) M B (\llbracket \rrbracket) (1)$ 
and rPar1:  $\bigwedge \Psi \Psi_Q P M xvec N P' A_Q Q A_P \Psi_P C.$ 
            $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P';$ 
            $extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$ 
            $extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$ 
            $\bigwedge C. Prop C (\Psi \otimes \Psi_Q) P M ((\nu * xvec) N \prec' P') A_P \Psi_P;$ 
            $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* xvec; A_P \#* N; A_P$ 
            $\#* P'; A_P \#* A_Q; A_P \#* \Psi_Q;$ 
            $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* xvec; A_Q \#* N; A_Q$ 
            $\#* P'; A_Q \#* \Psi_P;$ 
            $xvec \#* \Psi; xvec \#* P; xvec \#* Q; xvec \#* M; xvec \#* \Psi_P; xvec \#* \Psi_Q;$ 
            $A_P \#* C; A_Q \#* C; xvec \#* C \rrbracket \implies$ 
            $Prop C \Psi (P \parallel Q) M ((\nu * xvec) N \prec' (P' \parallel Q)) (A_P @ A_Q) (\Psi_P \otimes$ 
            $\Psi_Q)$ 
and rPar2:  $\bigwedge \Psi \Psi_P Q M xvec N Q' A_P P A_Q \Psi_Q C.$ 
            $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec Q';$ 
            $extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$ 
            $extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$ 
            $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q M ((\nu * xvec) N \prec' Q') A_Q \Psi_Q;$ 
            $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* M; A_P \#* xvec; A_P \#* N; A_P$ 
            $\#* Q'; A_P \#* A_Q; A_P \#* \Psi_Q;$ 
            $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* M; A_Q \#* xvec; A_Q \#* N; A_Q$ 
            $\#* Q'; A_Q \#* \Psi_P;$ 

```

$xvec \#* \Psi; xvec \#* P; xvec \#* Q; xvec \#* M; xvec \#* \Psi_P; xvec \#* \Psi_Q;$
 $A_P \#* C; A_Q \#* C; xvec \#* C \implies$
 $Prop C \Psi (P \parallel Q) M ((\nu*xvec)N \prec' (P \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes$
 $\Psi_Q)$
and $rBrComm1: \wedge \Psi \Psi_Q P M N P' A_P \Psi_P Q xvec Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
distinct $A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu*xvec)\langle N \rangle \prec Q'; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle;$ *distinct* $A_Q;$
distinct $xvec;$
 $\wedge C. Prop C (\Psi \otimes \Psi_P) Q M ((\nu*xvec)N \prec' Q') A_Q \Psi_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$
 $xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M \implies$
 $Prop C \Psi (P \parallel Q) M ((\nu*xvec)N \prec' (P' \parallel Q')) (A_P @ A_Q) (\Psi_P$
 $\otimes \Psi_Q)$
and $rBrComm2: \wedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle;$ *distinct* $A_P;$
 $\wedge C. Prop C (\Psi \otimes \Psi_Q) P M ((\nu*xvec)N \prec' P') A_P \Psi_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(N) \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
distinct $A_Q;$
distinct $xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q';$
 $A_Q \#* xvec; xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P;$
 $xvec \#* Q; A_P \#* C; A_Q \#* C; xvec \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M \implies$
 $Prop C \Psi (P \parallel Q) M ((\nu*xvec)N \prec' (P' \parallel Q')) (A_P @ A_Q) (\Psi_P$
 $\otimes \Psi_Q)$
and $rBrOpen: \wedge \Psi P M xvec yvec N P' x A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \mapsto_i M(\nu*(xvec@yvec))\langle N \rangle \prec P'; extractFrame P = \langle A_P,$
 $\Psi_P \rangle;$ *distinct* $A_P;$
 $\wedge C. Prop C \Psi P M ((\nu*(xvec@yvec))N \prec' P') A_P \Psi_P; x \in supp$
 $N; x \# \Psi; x \# M;$
 $x \# A_P; x \# xvec; x \# yvec; A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#*$
 $N; A_P \#* P';$
 $A_P \#* xvec; A_P \#* yvec;$
 $xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* \Psi_P;$
 $yvec \#* \Psi; yvec \#* P; yvec \#* M; A_P \#* C; x \# C; xvec \#* C; yvec$
 $\#* C \implies$
 $Prop C \Psi ((\nu x)P) M ((\nu*(xvec@x\#yvec))N \prec' P') (x\#A_P) \Psi_P$
and $rScope: \wedge \Psi P M xvec N P' x A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
distinct $A_P;$

$\wedge C. Prop C \Psi P M ((\nu * xvec)N \prec' P') A_P \Psi_P;$
 $x \# \Psi; x \# M; x \# xvec; x \# N; x \# A_P; A_P \#* \Psi; A_P \#* P;$
 $A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* xvec;$
 $xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* \Psi_P;$
 $A_P \#* C; x \# C; xvec \#* C] \implies$
 $Prop C \Psi ((\nu x)P) M ((\nu * xvec)N \prec' ((\nu x)P')) (x \# A_P) \Psi_P$
and *rBang*: $\wedge \Psi P M B A_P \Psi_P C.$
 $[[\Psi \triangleright P \parallel !P \mapsto RBrOut M B; guarded P; extractFrame P =$
 $\langle A_P, \Psi_P \rangle; distinct A_P;$
 $\wedge C. Prop C \Psi (P \parallel !P) M B A_P (\Psi_P \otimes \mathbf{1}); \Psi_P \simeq \mathbf{1}; supp \Psi_P$
 $= (\{\} :: name set);$
 $A_P \#* \Psi; A_P \#* P; A_P \#* M; A_P \#* C] \implies Prop C \Psi (!P) M$
 $B (\parallel) (\mathbf{1})$
shows $Prop C \Psi P M B A_P \Psi_P$
 $\langle proof \rangle$

lemma *tauFrameInduct*[*consumes 3, case-names cAlpha cCase cPar1 cPar2 cComm1 cComm2 cBrClose cScope cBang*]:

fixes $\Psi \quad :: 'b$
and $P \quad :: ('a, 'b, 'c) psi$
and $P' \quad :: ('a, 'b, 'c) psi$
and $Prop \quad :: 'f :: fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow$
 $('a, 'b, 'c) psi \Rightarrow name list \Rightarrow 'b \Rightarrow bool$
and $C \quad :: 'f :: fs-name$
assumes *Trans*: $\Psi \triangleright P \mapsto \tau \prec P'$
and *FrP*: $extractFrame P = \langle A_P, \Psi_P \rangle$
and *distinct* A_P
and *rAlpha*: $\wedge \Psi P P' A_P \Psi_P p C. [[A_P \#* \Psi; A_P \#* P; A_P \#* P'; A_P \#* (p \cdot$
 $A_P); A_P \#* C;$
 $set p \subseteq set A_P \times set (p \cdot A_P); distinctPerm p;$
 $Prop C \Psi P P' A_P \Psi_P] \implies Prop C \Psi P P' (p \cdot$
 $A_P) (p \cdot \Psi_P)$
and *rCase*: $\wedge \Psi P P' \varphi Cs A_P \Psi_P C. [[\Psi \triangleright P \mapsto \tau \prec P'; extractFrame P =$
 $\langle A_P, \Psi_P \rangle; distinct A_P; \wedge C. Prop C \Psi P P' A_P \Psi_P;$
 $(\varphi, P) \in set Cs; \Psi \vdash \varphi; guarded P; \Psi_P \simeq \mathbf{1};$
 $(supp \Psi_P) = (\{\} :: name set);$
 $A_P \#* \Psi; A_P \#* P; A_P \#* P'; A_P \#* C] \implies$
 $Prop C \Psi (Cases Cs) P' (\parallel) (\mathbf{1})$
and *rPar1*: $\wedge \Psi \Psi_Q P P' A_Q Q A_P \Psi_P C.$
 $[[\Psi \otimes \Psi_Q \triangleright P \mapsto \tau \prec P';$
 $extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $\wedge C. Prop C (\Psi \otimes \Psi_Q) P P' A_P \Psi_P;$
 $A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* P'; A_P \#* A_Q; A_P \#* \Psi_Q;$
 $A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* P'; A_Q \#* \Psi_P;$
 $A_P \#* C; A_Q \#* C] \implies$
 $Prop C \Psi (P \parallel Q) (P' \parallel Q) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and *rPar2*: $\wedge \Psi \Psi_P Q Q' A_P P A_Q \Psi_Q C.$

$\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \tau \prec Q' ;$
 $extractFrame P = \langle A_P, \Psi_P \rangle ; distinct A_P ;$
 $extractFrame Q = \langle A_Q, \Psi_Q \rangle ; distinct A_Q ;$
 $\bigwedge C. Prop C (\Psi \otimes \Psi_P) Q Q' A_Q \Psi_Q ;$
 $A_P \#* P ; A_P \#* Q ; A_P \#* \Psi ; A_P \#* Q' ; A_P \#* A_Q ; A_P \#* \Psi_Q ;$
 $A_Q \#* P ; A_Q \#* Q ; A_Q \#* \Psi ; A_Q \#* Q' ; A_Q \#* \Psi_P ;$
 $A_P \#* C ; A_Q \#* C \rrbracket \implies$
 $Prop C \Psi (P \parallel Q) (P \parallel Q') (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rComm1: \bigwedge \Psi \Psi_Q P M N P' A_P \Psi_P Q K xvec Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(N) \prec P' ; extractFrame P = \langle A_P, \Psi_P \rangle ;$
 $distinct A_P ;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * xvec) \langle N \rangle \prec Q' ; extractFrame Q = \langle A_Q,$
 $\Psi_Q \rangle ; distinct A_Q ;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K ; distinct xvec ;$
 $A_P \#* \Psi ; A_P \#* \Psi_Q ; A_P \#* P ; A_P \#* M ; A_P \#* N ; A_P \#* P' ;$
 $A_P \#* Q ; A_P \#* Q' ; A_P \#* A_Q ; A_P \#* xvec ; A_Q \#* \Psi ; A_Q \#* \Psi_P ;$
 $A_Q \#* P ; A_Q \#* N ; A_Q \#* P' ; A_Q \#* Q ; A_Q \#* K ; A_Q \#* Q' ;$
 $A_Q \#* xvec ; xvec \#* \Psi ; xvec \#* \Psi_P ; xvec \#* \Psi_Q ; xvec \#* P ; xvec \#*$
 $M ;$
 $xvec \#* Q ; xvec \#* K ; A_P \#* C ; A_Q \#* C ; xvec \#* C \rrbracket \implies$
 $Prop C \Psi (P \parallel Q) ((\nu * xvec)(P' \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rComm2: \bigwedge \Psi \Psi_Q P M xvec N P' A_P \Psi_P Q K Q' A_Q C.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec) \langle N \rangle \prec P' ; extractFrame P = \langle A_P,$
 $\Psi_P \rangle ; distinct A_P ;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(N) \prec Q' ; extractFrame Q = \langle A_Q, \Psi_Q \rangle ;$
 $distinct A_Q ;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K ; distinct xvec ;$
 $A_P \#* \Psi ; A_P \#* \Psi_Q ; A_P \#* P ; A_P \#* M ; A_P \#* N ; A_P \#* P' ;$
 $A_P \#* Q ; A_P \#* Q' ; A_P \#* A_Q ; A_P \#* xvec ; A_Q \#* \Psi ; A_Q \#* \Psi_P ;$
 $A_Q \#* P ; A_Q \#* N ; A_Q \#* P' ; A_Q \#* Q ; A_Q \#* K ; A_Q \#* Q' ;$
 $A_Q \#* xvec ; xvec \#* \Psi ; xvec \#* \Psi_P ; xvec \#* \Psi_Q ; xvec \#* P ; xvec \#*$
 $M ;$
 $xvec \#* Q ; xvec \#* K ; A_P \#* C ; A_Q \#* C ; xvec \#* C \rrbracket \implies$
 $Prop C \Psi (P \parallel Q) ((\nu * xvec)(P' \parallel Q')) (A_P @ A_Q) (\Psi_P \otimes \Psi_Q)$
and $rBrClose: \bigwedge \Psi P M xvec N P' A_P \Psi_P x C.$
 $\llbracket \Psi \triangleright P \mapsto \text{!}M(\nu * xvec) \langle N \rangle \prec P' ;$
 $x \in supp M ;$
 $extractFrame P = \langle A_P, \Psi_P \rangle ; distinct A_P ;$
 $A_P \#* \Psi ; A_P \#* P ; A_P \#* M ; A_P \#* N ; A_P \#* P' ; A_P \#* xvec ;$
 $distinct xvec ; xvec \#* \Psi ; xvec \#* \Psi_P ; xvec \#* P ;$
 $xvec \#* M ;$
 $x \# \Psi ; x \# xvec ; x \# A_P ;$
 $A_P \#* C ; xvec \#* C ; x \# C \rrbracket \implies$
 $Prop C \Psi ((\nu x)P) ((\nu x)((\nu * xvec)P')) (x \# A_P) \Psi_P$
and $rScope: \bigwedge \Psi P P' x A_P \Psi_P C.$
 $\llbracket \Psi \triangleright P \mapsto \tau \prec P' ; extractFrame P = \langle A_P, \Psi_P \rangle ; distinct A_P ;$
 $\bigwedge C. Prop C \Psi P P' A_P \Psi_P ; x \# \Psi ;$
 $x \# A_P ; A_P \#* \Psi ; A_P \#* P ; A_P \#* P' ;$
 $A_P \#* C ; x \# C \rrbracket \implies$

$$\text{and } rBang: \quad \text{Prop } C \Psi ((\nu x)P) ((\nu x)P') (x\#A_P) \Psi_P$$

$$\wedge \Psi P P' A_P \Psi_P C.$$

$$\llbracket \Psi \triangleright P \parallel !P \mapsto \tau \prec P'; \text{ guarded } P; \text{ extractFrame } P = \langle A_P, \Psi_P \rangle;$$

$$\text{distinct } A_P;$$

$$\wedge C. \text{ Prop } C \Psi (P \parallel !P) P' A_P (\Psi_P \otimes \mathbf{1}); \Psi_P \simeq \mathbf{1}; \text{ supp } \Psi_P =$$

$$(\{\}::\text{name set});$$

$$A_P \#* \Psi; A_P \#* P; A_P \#* P'; A_P \#* C \rrbracket \implies \text{Prop } C \Psi (!P) P'$$

$$(\square) (1)$$
shows $\text{Prop } C \Psi P P' A_P \Psi_P$

$$\langle \text{proof} \rangle$$

lemma *inputFreshDerivative:*

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $x :: \text{name}$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$
and $x \# P$
and $x \# N$

shows $x \# P'$
 $\langle \text{proof} \rangle$

lemma *brinputFreshDerivative:*

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $x :: \text{name}$

assumes $\Psi \triangleright P \mapsto_i M(N) \prec P'$
and $x \# P$
and $x \# N$

shows $x \# P'$
 $\langle \text{proof} \rangle$

lemma *inputFreshChainDerivative:*

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $xvec :: \text{name list}$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$
and $xvec \#* P$
and $xvec \#* N$

shows $xvec \#* P'$
<proof>

lemma *brinputFreshChainDerivative:*

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $xvec :: name list$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$
and $xvec \#* P$
and $xvec \#* N$

shows $xvec \#* P'$
<proof>

lemma *outputFreshDerivativeN:*

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $xvec :: name list$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $x :: name$

assumes $\Psi \triangleright P \mapsto M(\nu*xvec)(N) \prec P'$
and $xvec \#* M$
and *distinct* $xvec$
and $x \# P$
and $x \# xvec$

shows $x \# N$
<proof>

lemma *broutputFreshDerivativeN:*

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $xvec :: name list$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $x :: name$

assumes $\Psi \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P'$
and $xvec \#^* M$
and $distinct\ xvec$
and $x \# P$
and $x \# xvec$

shows $x \# N$
 $\langle proof \rangle$

lemma *outputFreshDerivativeP:*

fixes $\Psi \quad :: 'b$
and $P \quad :: ('a, 'b, 'c)\ psi$
and $M \quad :: 'a$
and $xvec \quad :: name\ list$
and $N \quad :: 'a$
and $P' \quad :: ('a, 'b, 'c)\ psi$
and $x \quad :: name$

assumes $\Psi \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P'$
and $xvec \#^* M$
and $distinct\ xvec$
and $x \# P$
and $x \# xvec$

shows $x \# P'$
 $\langle proof \rangle$

lemma *broutputFreshDerivativeP:*

fixes $\Psi \quad :: 'b$
and $P \quad :: ('a, 'b, 'c)\ psi$
and $M \quad :: 'a$
and $xvec \quad :: name\ list$
and $N \quad :: 'a$
and $P' \quad :: ('a, 'b, 'c)\ psi$
and $x \quad :: name$

assumes $\Psi \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P'$
and $xvec \#^* M$
and $distinct\ xvec$
and $x \# P$
and $x \# xvec$

shows $x \# P'$
 $\langle proof \rangle$

lemma *outputFreshDerivative:*

fixes $\Psi \quad :: 'b$
and $P \quad :: ('a, 'b, 'c)\ psi$
and $M \quad :: 'a$

```

and  $xvec$  :: name list
and  $N$     :: 'a
and  $P'$    :: ('a, 'b, 'c) psi
and  $x$     :: name

assumes  $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and distinct  $xvec$ 
and  $x \# P$ 
and  $x \# xvec$ 

shows  $x \# N$ 
and  $x \# P'$ 
  <proof>

lemma broutputFreshDerivative:
fixes  $\Psi$    :: 'b
and  $P$      :: ('a, 'b, 'c) psi
and  $M$      :: 'a
and  $xvec$   :: name list
and  $N$      :: 'a
and  $P'$     :: ('a, 'b, 'c) psi
and  $x$      :: name

assumes  $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and distinct  $xvec$ 
and  $x \# P$ 
and  $x \# xvec$ 

shows  $x \# N$ 
and  $x \# P'$ 
  <proof>

lemma outputFreshChainDerivative:
fixes  $\Psi$    :: 'b
and  $P$      :: ('a, 'b, 'c) psi
and  $M$      :: 'a
and  $xvec$   :: name list
and  $N$      :: 'a
and  $P'$     :: ('a, 'b, 'c) psi
and  $yvec$   :: name list

assumes  $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and distinct  $xvec$ 
and  $yvec \#^* P$ 
and  $yvec \#^* xvec$ 

```


shows $yvec \#* N$
and $yvec \#* P'$
 $\langle proof \rangle$

lemma *broutputFreshChainDerivative*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $xvec :: name list$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $yvec :: name list$

assumes $\Psi \triangleright P \mapsto_{\downarrow} M(\nu*xvec)\langle N \rangle \prec P'$
and $xvec \#* M$
and *distinct* $xvec$
and $yvec \#* P$
and $yvec \#* xvec$

shows $yvec \#* N$
and $yvec \#* P'$
 $\langle proof \rangle$

lemma *tauFreshDerivative*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $P' :: ('a, 'b, 'c) psi$
and $x :: name$

assumes $\Psi \triangleright P \mapsto_{\tau} \prec P'$
and $x \# P$

shows $x \# P'$
 $\langle proof \rangle$

lemma *tauFreshChainDerivative*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $xvec :: name list$

assumes $\Psi \triangleright P \mapsto_{\tau} \prec P'$
and $xvec \#* P$

shows $xvec \#* P'$
 $\langle proof \rangle$

lemma *freeFreshDerivative*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c)$ *psi*
and $\alpha :: 'a$ *action*
and $P' :: ('a, 'b, 'c)$ *psi*
and $x :: \text{name}$

assumes $\Psi \triangleright P \mapsto \alpha \prec P'$

and $bn\ \alpha \#* \text{subject}\ \alpha$
and $distinct(bn\ \alpha)$
and $x \# \alpha$
and $x \# P$

shows $x \# P'$

<proof>

lemma *freeFreshChainDerivative*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c)$ *psi*
and $\alpha :: 'a$ *action*
and $P' :: ('a, 'b, 'c)$ *psi*
and $xvec :: \text{name list}$

assumes $\Psi \triangleright P \mapsto \alpha \prec P'$

and $bn\ \alpha \#* \text{subject}\ \alpha$
and $distinct(bn\ \alpha)$
and $xvec \#* P$
and $xvec \#* \alpha$

shows $xvec \#* P'$

<proof>

lemma *Input*:

fixes $\Psi :: 'b$
and $M :: 'a$
and $K :: 'a$
and $xvec :: \text{name list}$
and $N :: 'a$
and $Tvec :: 'a \text{ list}$

assumes $\Psi \vdash M \leftrightarrow K$

and $distinct\ xvec$
and $set\ xvec \subseteq \text{supp}\ N$
and $length\ xvec = length\ Tvec$

shows $\Psi \triangleright M(\lambda x. xvec\ N).P \mapsto K(\lambda N[xvec ::= Tvec]) \prec P[xvec ::= Tvec]$

<proof>

lemma *BrInput*:

```

fixes  $\Psi$   :: 'b
and  $M$     :: 'a
and  $K$     :: 'a
and  $xvec$  :: name list
and  $N$     :: 'a
and  $Tvec$  :: 'a list

assumes  $\Psi \vdash K \succeq M$ 
and distinct  $xvec$ 
and set  $xvec \subseteq \text{supp } N$ 
and length  $xvec = \text{length } Tvec$ 

shows  $\Psi \triangleright M(\lambda * xvec N).P \mapsto_i K(N[xvec ::= Tvec]) \prec P[xvec ::= Tvec]$ 
<proof>

lemma residualAlpha:
fixes  $p$  :: name prm
and  $\alpha$   :: 'a action
and  $P$   :: ('a, 'b, 'c) psi

assumes  $bn(p \cdot \alpha) \#* \text{object } \alpha$ 
and  $bn(p \cdot \alpha) \#* P$ 
and  $bn \alpha \#* \text{subject } \alpha$ 
and  $bn(p \cdot \alpha) \#* \text{subject } \alpha$ 
and set  $p \subseteq \text{set}(bn \alpha) \times \text{set}(bn(p \cdot \alpha))$ 

shows  $\alpha \prec P = (p \cdot \alpha) \prec (p \cdot P)$ 
<proof>

lemma Par1:
fixes  $\Psi$   :: 'b
and  $\Psi_Q$   :: 'b
and  $P$     :: ('a, 'b, 'c) psi
and  $\alpha$    :: 'a action
and  $P'$   :: ('a, 'b, 'c) psi
and  $A_Q$   :: name list
and  $Q$     :: ('a, 'b, 'c) psi

assumes Trans:  $\Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P'$ 
and extractFrame  $Q = \langle A_Q, \Psi_Q \rangle$ 
and  $bn \alpha \#* Q$ 
and  $A_Q \#* \Psi$ 
and  $A_Q \#* P$ 
and  $A_Q \#* \alpha$ 

shows  $\Psi \triangleright P \parallel Q \mapsto \alpha \prec (P' \parallel Q)$ 
<proof>

lemma Par2:

```

fixes Ψ :: 'b
and Ψ_P :: 'b
and Q :: ('a, 'b, 'c) psi
and α :: 'a action
and Q' :: ('a, 'b, 'c) psi
and A_P :: name list
and P :: ('a, 'b, 'c) psi

assumes *Trans*: $\Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q'$
and *extractFrame* $P = \langle A_P, \Psi_P \rangle$
and $bn \ \alpha \ \#^* P$
and $A_P \ \#^* \Psi$
and $A_P \ \#^* Q$
and $A_P \ \#^* \alpha$

shows $\Psi \triangleright P \parallel Q \mapsto \alpha \prec (P \parallel Q')$
<proof>

lemma *Open*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and $yvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and x :: name

assumes *Trans*: $\Psi \triangleright P \mapsto M(\nu^*(xvec@yvec))\langle N \rangle \prec P'$
and $x \in supp \ N$
and $x \ \# \ \Psi$
and $x \ \# \ M$
and $x \ \# \ xvec$
and $x \ \# \ yvec$

shows $\Psi \triangleright (\nu x)P \mapsto M(\nu^*(xvec@x\#yvec))\langle N \rangle \prec P'$
<proof>

lemma *BrOpen*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and $yvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and x :: name

assumes *Trans*: $\Psi \triangleright P \mapsto_i M(\nu^*(xvec@yvec))\langle N \rangle \prec P'$

and $x \in \text{supp } N$
and $x \# \Psi$
and $x \# M$
and $x \# \text{xvec}$
and $x \# \text{yvec}$

shows $\Psi \triangleright (\nu x)P \mapsto_i M(\nu*(\text{xvec}@x\#\text{yvec}))\langle N \rangle \prec P'$
 $\langle \text{proof} \rangle$

lemma *Scope*:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) \text{ psi}$
and α $:: 'a \text{ action}$
and P' $:: ('a, 'b, 'c) \text{ psi}$
and x $:: \text{ name}$

assumes $\Psi \triangleright P \mapsto_\alpha \prec P'$
and $x \# \Psi$
and $x \# \alpha$

shows $\Psi \triangleright (\nu x)P \mapsto_\alpha \prec (\nu x)P'$
 $\langle \text{proof} \rangle$

lemma *inputSwapFrameSubject*:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) \text{ psi}$
and M $:: 'a$
and N $:: 'a$
and P' $:: ('a, 'b, 'c) \text{ psi}$
and x $:: \text{ name}$
and y $:: \text{ name}$

assumes $\Psi \triangleright P \mapsto M\langle N \rangle \prec P'$
and $x \# P$
and $y \# P$

shows $([(x, y)] \cdot \Psi) \triangleright P \mapsto ([(x, y)] \cdot M)\langle N \rangle \prec P'$
 $\langle \text{proof} \rangle$

lemma *brinputSwapFrameSubject*:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) \text{ psi}$
and M $:: 'a$
and N $:: 'a$
and P' $:: ('a, 'b, 'c) \text{ psi}$
and x $:: \text{ name}$
and y $:: \text{ name}$

assumes $\Psi \triangleright P \mapsto_i M\langle N \rangle \prec P'$

and $x \# P$
and $y \# P$

shows $([(x, y)] \cdot \Psi) \triangleright P \mapsto \iota([(x, y)] \cdot M)(\downarrow N) \prec P'$
<proof>

lemma *inputPermFrameSubject*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $p :: \text{ name prm}$
and $Xs :: \text{ name set}$
and $Ys :: \text{ name set}$

assumes $\Psi \triangleright P \mapsto M(\downarrow N) \prec P'$
and $S: \text{ set } p \subseteq Xs \times Ys$
and $Xs \#* P$
and $Ys \#* P$

shows $(p \cdot \Psi) \triangleright P \mapsto (p \cdot M)(\downarrow N) \prec P'$
<proof>

lemma *brinputPermFrameSubject*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $p :: \text{ name prm}$
and $Xs :: \text{ name set}$
and $Ys :: \text{ name set}$

assumes $\Psi \triangleright P \mapsto \iota M(\downarrow N) \prec P'$
and $S: \text{ set } p \subseteq Xs \times Ys$
and $Xs \#* P$
and $Ys \#* P$

shows $(p \cdot \Psi) \triangleright P \mapsto \iota(p \cdot M)(\downarrow N) \prec P'$
<proof>

lemma *inputSwapSubject*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $x :: \text{ name}$

and $y :: name$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$
and $x \# P$
and $y \# P$
and $x \# \Psi$
and $y \# \Psi$

shows $\Psi \triangleright P \mapsto ((x, y) \cdot M)(N) \prec P'$
<proof>

lemma *brinputSwapSubject*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $x :: name$
and $y :: name$

assumes $\Psi \triangleright P \mapsto \iota M(N) \prec P'$
and $x \# P$
and $y \# P$
and $x \# \Psi$
and $y \# \Psi$

shows $\Psi \triangleright P \mapsto \iota((x, y) \cdot M)(N) \prec P'$
<proof>

lemma *inputPermSubject*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $p :: name prm$
and $Xs :: name set$
and $Ys :: name set$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$
and $S: set\ p \subseteq Xs \times Ys$
and $Xs \#* P$
and $Ys \#* P$
and $Xs \#* \Psi$
and $Ys \#* \Psi$

shows $\Psi \triangleright P \mapsto (p \cdot M)(N) \prec P'$
<proof>

lemma *brinputPermSubject*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $p :: \text{ name prm}$
and $Xs :: \text{ name set}$
and $Ys :: \text{ name set}$

assumes $\Psi \triangleright P \mapsto \iota M(N) \prec P'$

and $S: \text{ set } p \subseteq Xs \times Ys$
and $Xs \#* P$
and $Ys \#* P$
and $Xs \#* \Psi$
and $Ys \#* \Psi$

shows $\Psi \triangleright P \mapsto \iota(p \cdot M)(N) \prec P'$
<proof>

lemma *inputSwapFrame*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $x :: \text{ name}$
and $y :: \text{ name}$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$

and $x \# P$
and $y \# P$
and $x \# M$
and $y \# M$

shows $[(x, y)] \cdot \Psi \triangleright P \mapsto M(N) \prec P'$
<proof>

lemma *brinputSwapFrame*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $x :: \text{ name}$
and $y :: \text{ name}$

assumes $\Psi \triangleright P \mapsto \iota M(N) \prec P'$

and $x \# P$

and $y \# P$
and $x \# M$
and $y \# M$

shows $([(x, y)] \cdot \Psi) \triangleright P \mapsto \iota M(N) \prec P'$
 $\langle proof \rangle$

lemma *inputPermFrame*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $p :: name prm$
and $Xs :: name set$
and $Ys :: name set$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$

and $S: set p \subseteq Xs \times Ys$
and $Xs \#* P$
and $Ys \#* P$
and $Xs \#* M$
and $Ys \#* M$

shows $(p \cdot \Psi) \triangleright P \mapsto M(N) \prec P'$
 $\langle proof \rangle$

lemma *brinputPermFrame*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $p :: name prm$
and $Xs :: name set$
and $Ys :: name set$

assumes $\Psi \triangleright P \mapsto \iota M(N) \prec P'$

and $S: set p \subseteq Xs \times Ys$
and $Xs \#* P$
and $Ys \#* P$
and $Xs \#* M$
and $Ys \#* M$

shows $(p \cdot \Psi) \triangleright P \mapsto \iota M(N) \prec P'$
 $\langle proof \rangle$

lemma *inputAlpha*:

fixes $\Psi :: 'b$

and P :: ('a, 'b, 'c) psi
and M :: 'a
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and p :: name prm
and $xvec$:: name list

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$
and $set\ p \subseteq (set\ xvec) \times (set\ (p \cdot xvec))$
and $distinctPerm\ p$
and $xvec \#* P$
and $(p \cdot xvec) \#* P$

shows $\Psi \triangleright P \mapsto M((p \cdot N)) \prec (p \cdot P')$
<proof>

lemma *brinputAlpha*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and p :: name prm
and $xvec$:: name list

assumes $\Psi \triangleright P \mapsto_i M(N) \prec P'$
and $set\ p \subseteq (set\ xvec) \times (set\ (p \cdot xvec))$
and $distinctPerm\ p$
and $xvec \#* P$
and $(p \cdot xvec) \#* P$

shows $\Psi \triangleright P \mapsto_i M((p \cdot N)) \prec (p \cdot P')$
<proof>

lemma *frameFresh[dest]*:

fixes x :: name
and A_F :: name list
and Ψ_F :: 'b

assumes $x \# A_F$
and $x \# \langle A_F, \Psi_F \rangle$

shows $x \# \Psi_F$
<proof>

lemma *outputSwapFrameSubject*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a

```

and  $xvec$  :: name list
and  $N$     :: 'a
and  $x$     :: name
and  $y$     :: name

assumes  $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and  $x \# P$ 
and  $y \# P$ 

shows  $([(x, y)] \cdot \Psi) \triangleright P \mapsto ([(x, y)] \cdot M)(\nu*xvec)\langle N \rangle \prec P'$ 
  <proof>

lemma broutputSwapFrameSubject:
  fixes  $\Psi$     :: 'b
    and  $P$     :: ('a, 'b, 'c) psi
    and  $M$     :: 'a
    and  $xvec$  :: name list
    and  $N$     :: 'a
    and  $x$     :: name
    and  $y$     :: name

assumes  $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and  $x \# P$ 
and  $y \# P$ 

shows  $([(x, y)] \cdot \Psi) \triangleright P \mapsto_i ([(x, y)] \cdot M)(\nu*xvec)\langle N \rangle \prec P'$ 
  <proof>

lemma outputPermFrameSubject:
  fixes  $\Psi$     :: 'b
    and  $P$     :: ('a, 'b, 'c) psi
    and  $M$     :: 'a
    and  $xvec$  :: name list
    and  $N$     :: 'a
    and  $P'$    :: ('a, 'b, 'c) psi
    and  $p$     :: name prm
    and  $yvec$  :: name list
    and  $zvec$  :: name list

assumes  $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $S$ : set  $p \subseteq$  set  $yvec \times$  set  $zvec$ 
and  $yvec \#^* P$ 
and  $zvec \#^* P$ 

shows  $(p \cdot \Psi) \triangleright P \mapsto (p \cdot M)(\nu*xvec)\langle N \rangle \prec P'$ 
  <proof>

```

lemma *broutputPermFrameSubject*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and p :: name prm
and $yvec$:: name list
and $zvec$:: name list

assumes $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$

and S : set $p \subseteq$ set $yvec \times$ set $zvec$
and $yvec \#^* P$
and $zvec \#^* P$

shows $(p \cdot \Psi) \triangleright P \mapsto_i (p \cdot M)(\nu*xvec)\langle N \rangle \prec P'$
<proof>

lemma *outputSwapSubject*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and B :: ('a, 'b, 'c) boundOutput
and x :: name
and y :: name

assumes $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$

and $xvec \#^* M$
and $x \# P$
and $y \# P$
and $x \# \Psi$
and $y \# \Psi$

shows $\Psi \triangleright P \mapsto_i ((x, y) \cdot M)(\nu*xvec)\langle N \rangle \prec P'$
<proof>

lemma *broutputSwapSubject*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and B :: ('a, 'b, 'c) boundOutput
and x :: name
and y :: name

assumes $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$

and $xvec \#^* M$
and $x \# P$
and $y \# P$

and $x \# \Psi$
and $y \# \Psi$

shows $\Psi \triangleright P \mapsto_i([(x, y)] \cdot M)(\nu*xvec)\langle N \rangle \prec P'$
 $\langle proof \rangle$

lemma *outputPermSubject*:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) psi$
and M $:: 'a$
and B $:: ('a, 'b, 'c) boundOutput$
and p $:: name prm$
and $yvec$ $:: name list$
and $zvec$ $:: name list$

assumes $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$

and $S: set\ p \subseteq set\ yvec \times set\ zvec$
and $yvec \#* P$
and $zvec \#* P$
and $yvec \#* \Psi$
and $zvec \#* \Psi$

shows $\Psi \triangleright P \mapsto (p \cdot M)(\nu*xvec)\langle N \rangle \prec P'$
 $\langle proof \rangle$

lemma *broutputPermSubject*:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) psi$
and M $:: 'a$
and B $:: ('a, 'b, 'c) boundOutput$
and p $:: name prm$
and $yvec$ $:: name list$
and $zvec$ $:: name list$

assumes $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$

and $S: set\ p \subseteq set\ yvec \times set\ zvec$
and $yvec \#* P$
and $zvec \#* P$
and $yvec \#* \Psi$
and $zvec \#* \Psi$

shows $\Psi \triangleright P \mapsto_i (p \cdot M)(\nu*xvec)\langle N \rangle \prec P'$
 $\langle proof \rangle$

lemma *outputSwapFrame*:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) psi$
and M $:: 'a$
and B $:: ('a, 'b, 'c) boundOutput$

```

and  $x$   :: name
and  $y$   :: name

assumes  $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and  $x \# P$ 
and  $y \# P$ 
and  $x \# M$ 
and  $y \# M$ 

shows  $([(x, y)] \cdot \Psi) \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
 $\langle proof \rangle$ 

lemma broutputSwapFrame:
fixes  $\Psi$   :: 'b'
and  $P$     :: ('a, 'b', 'c') psi
and  $M$     :: 'a'
and  $B$     :: ('a, 'b', 'c') boundOutput
and  $x$     :: name
and  $y$     :: name

assumes  $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $xvec \#^* M$ 
and  $x \# P$ 
and  $y \# P$ 
and  $x \# M$ 
and  $y \# M$ 

shows  $([(x, y)] \cdot \Psi) \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$ 
 $\langle proof \rangle$ 

lemma outputPermFrame:
fixes  $\Psi$   :: 'b'
and  $P$     :: ('a, 'b', 'c') psi
and  $M$     :: 'a'
and  $B$     :: ('a, 'b', 'c') boundOutput
and  $p$     :: name prm
and  $yvec$  :: name list
and  $zvec$  :: name list

assumes  $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
and  $S$ : set  $p \subseteq \text{set } yvec \times \text{set } zvec$ 
and  $yvec \#^* P$ 
and  $zvec \#^* P$ 
and  $yvec \#^* M$ 
and  $zvec \#^* M$ 

shows  $(p \cdot \Psi) \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$ 
 $\langle proof \rangle$ 

```

lemma *broutputPermFrame*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) *psi*
and M :: 'a
and B :: ('a, 'b, 'c) *boundOutput*
and p :: *name prm*
and $yvec$:: *name list*
and $zvec$:: *name list*

assumes $\Psi \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$

and S : *set* $p \subseteq \text{set } yvec \times \text{set } zvec$
and $yvec \#^* P$
and $zvec \#^* P$
and $yvec \#^* M$
and $zvec \#^* M$

shows $(p \cdot \Psi) \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'$
<proof>

lemma *Comm1*:

fixes Ψ :: 'b
and Ψ_Q :: 'b
and P :: ('a, 'b, 'c) *psi*
and M :: 'a
and N :: 'a
and P' :: ('a, 'b, 'c) *psi*
and A_P :: *name list*
and Ψ_P :: 'b
and Q :: ('a, 'b, 'c) *psi*
and K :: 'a
and $xvec$:: *name list*
and Q' :: ('a, 'b, 'c) *psi*
and A_Q :: *name list*

assumes $\Psi \otimes \Psi_Q \triangleright P \mapsto M(N) \prec P'$

and *extractFrame* $P = \langle A_P, \Psi_P \rangle$
and $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu*xvec)\langle N \rangle \prec Q'$
and *extractFrame* $Q = \langle A_Q, \Psi_Q \rangle$
and $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K$
and $A_P \#^* \Psi$
and $A_P \#^* P$
and $A_P \#^* Q$
and $A_P \#^* M$
and $A_P \#^* A_Q$
and $A_Q \#^* \Psi$
and $A_Q \#^* P$
and $A_Q \#^* Q$
and $A_Q \#^* K$

```

and   xvec #* P

shows  $\Psi \triangleright P \parallel Q \mapsto \tau \prec (\nu * xvec)(P' \parallel Q')$ 
 $\langle proof \rangle$ 

lemma Comm2:
  fixes  $\Psi$    :: 'b
    and  $\Psi_Q$   :: 'b
    and P     :: ('a, 'b, 'c) psi
    and M     :: 'a
    and xvec  :: name list
    and N     :: 'a
    and P'    :: ('a, 'b, 'c) psi
    and AP   :: name list
    and  $\Psi_P$   :: 'b
    and Q     :: ('a, 'b, 'c) psi
    and K     :: 'a
    and Q'   :: ('a, 'b, 'c) psi
    and AQ   :: name list

assumes  $\Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'$ 
  and extractFrame P =  $\langle A_P, \Psi_P \rangle$ 
  and  $\Psi \otimes \Psi_P \triangleright Q \mapsto K\langle N \rangle \prec Q'$ 
  and extractFrame Q =  $\langle A_Q, \Psi_Q \rangle$ 
  and  $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K$ 
  and AP #*  $\Psi$ 
  and AP #* P
  and AP #* Q
  and AP #* M
  and AP #* AQ
  and AQ #*  $\Psi$ 
  and AQ #* P
  and AQ #* Q
  and AQ #* K
  and xvec #* Q

shows  $\Psi \triangleright P \parallel Q \mapsto \tau \prec (\nu * xvec)(P' \parallel Q')$ 
 $\langle proof \rangle$ 

lemma BrMerge:
  fixes  $\Psi$    :: 'b
    and  $\Psi_Q$   :: 'b
    and P     :: ('a, 'b, 'c) psi
    and M     :: 'a
    and N     :: 'a
    and P'    :: ('a, 'b, 'c) psi
    and AP   :: name list
    and  $\Psi_P$   :: 'b
    and Q     :: ('a, 'b, 'c) psi

```



```

    and Q' :: ('a, 'b, 'c) psi
    and A_Q :: name list

assumes  $\Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P'$ 
    and extractFrame P =  $\langle A_P, \Psi_P \rangle$ 
    and  $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(N) \prec Q'$ 
    and extractFrame Q =  $\langle A_Q, \Psi_Q \rangle$ 
    and A_P #*  $\Psi$ 
    and A_P #* P
    and A_P #* Q
    and A_P #* M
    and A_P #* A_Q
    and A_Q #*  $\Psi$ 
    and A_Q #* P
    and A_Q #* Q
    and A_Q #* M

shows  $\Psi \triangleright P \parallel Q \mapsto_i M(N) \prec (P' \parallel Q')$ 
<proof>

lemma BrComm1:
    fixes  $\Psi$  :: 'b
    and  $\Psi_Q$  :: 'b
    and P :: ('a, 'b, 'c) psi
    and M :: 'a
    and N :: 'a
    and P' :: ('a, 'b, 'c) psi
    and A_P :: name list
    and  $\Psi_P$  :: 'b
    and Q :: ('a, 'b, 'c) psi
    and xvec :: name list
    and Q' :: ('a, 'b, 'c) psi
    and A_Q :: name list

assumes  $\Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P'$ 
    and extractFrame P =  $\langle A_P, \Psi_P \rangle$ 
    and  $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec Q'$ 
    and extractFrame Q =  $\langle A_Q, \Psi_Q \rangle$ 
    and A_P #*  $\Psi$ 
    and A_P #* P
    and A_P #* Q
    and A_P #* M
    and A_P #* A_Q
    and A_Q #*  $\Psi$ 
    and A_Q #* P
    and A_Q #* Q
    and A_Q #* M
    and xvec #* P

```

shows $\Psi \triangleright P \parallel Q \mapsto_{iM}(\nu * xvec)\langle N \rangle \prec (P' \parallel Q')$
<proof>

lemma *BrComm2*:

fixes Ψ :: 'b
and Ψ_Q :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and A_P :: name list
and Ψ_P :: 'b
and Q :: ('a, 'b, 'c) psi
and Q' :: ('a, 'b, 'c) psi
and A_Q :: name list

assumes $\Psi \otimes \Psi_Q \triangleright P \mapsto_{iM}(\nu * xvec)\langle N \rangle \prec P'$

and $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $\Psi \otimes \Psi_P \triangleright Q \mapsto_{iM}\langle N \rangle \prec Q'$
and $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle$
and $A_P \#* \Psi$
and $A_P \#* P$
and $A_P \#* Q$
and $A_P \#* M$
and $A_P \#* A_Q$
and $A_Q \#* \Psi$
and $A_Q \#* P$
and $A_Q \#* Q$
and $A_Q \#* M$
and $xvec \#* Q$

shows $\Psi \triangleright P \parallel Q \mapsto_{iM}(\nu * xvec)\langle N \rangle \prec (P' \parallel Q')$
<proof>

lemma *BrClose*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and x :: name

assumes $\Psi \triangleright P \mapsto_{iM}(\nu * xvec)\langle N \rangle \prec P'$

and $x \in supp\ M$
and $x \# \Psi$

shows $\Psi \triangleright (\nu x)P \mapsto_{iM} \tau \prec (\nu x)((\nu * xvec)P')$

$\langle \text{proof} \rangle$

lemma *semanticsCasesAux*[*consumes 1, case-names cInput cBrInput cOutput cBrOutput cCase cPar1 cPar2 cComm1 cComm2 cBrMerge cBrComm1 cBrComm2 cBrClose cOpen cBrOpen cScope cBang*]:

fixes $cP :: ('a, 'b, 'c) \text{ psi}$
and $cRs :: ('a, 'b, 'c) \text{ residual}$
and $C :: 'f::\text{fs-name}$
and $x :: \text{name}$

assumes $\Psi \triangleright cP \mapsto cRs$

and $rInput: \bigwedge M K \text{ xvec } N \text{ Tvec } P. \llbracket cP = M(\lambda * \text{xvec } N).P; cRs = K((N[\text{xvec}::=\text{Tvec}])) \rrbracket$
 $\prec P[\text{xvec}::=\text{Tvec}];$

$\Psi \vdash M \leftrightarrow K; \text{distinct } \text{xvec}; \text{set } \text{xvec} \subseteq \text{supp } N;$

$\text{length } \text{xvec} = \text{length } \text{Tvec};$

$\text{xvec} \#* \text{Tvec}; \text{xvec} \#* \Psi; \text{xvec} \#* M; \text{xvec} \#* K;$

$\text{xvec} \#* C \rrbracket \implies \text{Prop}$

and $rBrInput: \bigwedge M K \text{ xvec } N \text{ Tvec } P. \llbracket cP = M(\lambda * \text{xvec } N).P; cRs = \iota K((N[\text{xvec}::=\text{Tvec}])) \rrbracket$
 $\prec P[\text{xvec}::=\text{Tvec}];$

$\Psi \vdash K \succeq M; \text{distinct } \text{xvec}; \text{set } \text{xvec} \subseteq \text{supp } N;$

$\text{length } \text{xvec} = \text{length } \text{Tvec};$

$\text{xvec} \#* \text{Tvec}; \text{xvec} \#* \Psi; \text{xvec} \#* M; \text{xvec} \#* K;$

$\text{xvec} \#* C \rrbracket \implies \text{Prop}$

and $rOutput: \bigwedge M K N P. \llbracket cP = M\langle N \rangle.P; cRs = K\langle N \rangle \prec P; \Psi \vdash M \leftrightarrow K \rrbracket$
 $\implies \text{Prop}$

and $rBrOutput: \bigwedge M K N P. \llbracket cP = M\langle N \rangle.P; cRs = \iota K\langle N \rangle \prec P; \Psi \vdash M \preceq K \rrbracket \implies \text{Prop}$

and $rCase: \bigwedge Cs P \varphi. \llbracket cP = \text{Cases } Cs; \Psi \triangleright P \mapsto cRs; (\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \implies \text{Prop}$

and $rPar1: \bigwedge \Psi_Q P \alpha P' Q A_Q. \llbracket cP = P \parallel Q; cRs = \alpha \prec (P' \parallel Q);$

$(\Psi \otimes \Psi_Q) \triangleright P \mapsto (\alpha \prec P'); \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct}$

$A_Q;$

$A_Q \#* P; A_Q \#* Q; A_Q \#* \Psi; A_Q \#* \alpha; A_Q \#* C; A_Q \#* P'; \text{bn } \alpha \#*$
 $\Psi; \text{bn } \alpha \#* \Psi_Q;$

$\text{bn } \alpha \#* Q; \text{bn } \alpha \#* P; \text{bn } \alpha \#* \text{subject } \alpha; \text{bn } \alpha \#* C; \text{distinct}(\text{bn } \alpha) \rrbracket$

\implies

Prop

and $rPar2: \bigwedge \Psi_P Q \alpha Q' P A_P. \llbracket cP = P \parallel Q; cRs = \alpha \prec (P \parallel Q');$

$(\Psi \otimes \Psi_P) \triangleright Q \mapsto \alpha \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct}$

$A_P;$

$A_P \#* P; A_P \#* Q; A_P \#* \Psi; A_P \#* \alpha; A_P \#* C;$

$A_P \#* Q'; \text{bn } \alpha \#* \Psi; \text{bn } \alpha \#* \Psi_P; \text{bn } \alpha \#* P; \text{bn } \alpha \#* Q; \text{bn } \alpha \#* \text{subject}$

$\alpha; \text{bn } \alpha \#* C; \text{distinct}(\text{bn } \alpha) \rrbracket \implies \text{Prop}$

and $rComm1: \bigwedge \Psi_Q P M N P' A_P \Psi_P Q K \text{ xvec } Q' A_Q.$

$\llbracket cP = P \parallel Q; cRs = \tau \prec (\nu * \text{xvec})P' \parallel Q';$

$\Psi \otimes \Psi_Q \triangleright P \mapsto M\langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$

$\text{distinct } A_P;$

$\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * \text{xvec})\langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q,$

Ψ_Q); *distinct* A_Q ;
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#*$
 $M; A_P \#* N;$
 $A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi;$
 $A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; A_Q$
 $\#* xvec;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#*$
 $Q;$
 $xvec \#* K; A_P \#* C; A_Q \#* C; xvec \#* C; \text{distinct } xvec \implies Prop$
and *rComm2*: $\bigwedge \Psi_Q P M xvec N P' A_P \Psi_P Q K Q' A_Q.$
 $\llbracket cP = P \parallel Q; cRs = \tau \prec (\nu * xvec) P' \parallel Q' \rrbracket;$
 $\Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec) \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P,$
 $\Psi_P \rangle;$ *distinct* A_P ;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_Q ;
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#*$
 $M; A_P \#* N;$
 $A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* xvec; A_Q \#* \Psi;$
 $A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; A_Q$
 $\#* xvec;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#* M; xvec \#*$
 $Q;$
 $xvec \#* K; A_P \#* C; A_Q \#* C; xvec \#* C; \text{distinct } xvec \implies Prop$
and *rBrMerge*: $\bigwedge \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q.$
 $\llbracket cP = (P \parallel Q); cRs = \imath M \langle N \rangle \prec (P' \parallel Q') \rrbracket;$
 $\Psi \otimes \Psi_Q \triangleright P \mapsto \imath M \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
distinct A_P ;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_Q ;
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* M; A_P \#* A_Q;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_Q \#* M; A_P \#* C; A_Q \#* C \implies Prop$
and *rBrComm1*: $\bigwedge \Psi_Q P M N P' A_P \Psi_P Q xvec Q' A_Q.$
 $\llbracket cP = P \parallel Q; cRs = \imath M(\nu * xvec) \langle N \rangle \prec (P' \parallel Q') \rrbracket;$
 $\Psi \otimes \Psi_Q \triangleright P \mapsto \imath M \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
distinct A_P ;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\nu * xvec) \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q,$
 $\Psi_Q \rangle;$ *distinct* A_Q ;
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N;$
 $A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* M; A_P \#* A_Q; A_P \#* xvec;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P;$
 $A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* M; A_Q \#* Q; A_Q \#* Q'; A_Q$
 $\#* xvec;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* \Psi_Q; xvec \#* P; xvec \#* Q; xvec \#*$
 $M;$
 $A_P \#* C; A_Q \#* C; xvec \#* C; \text{distinct } xvec \implies Prop$

and $rBrComm2$: $\bigwedge \Psi_Q P M xvec N P' A_P \Psi_P Q Q' A_Q$.
 $\llbracket cP = P \parallel Q; cRs = \text{i}M(\nu^*xvec)\langle N \rangle \prec (P' \parallel Q') \rrbracket$;
 $\Psi \otimes \Psi_Q \triangleright P \mapsto \text{i}M(\nu^*xvec)\langle N \rangle \prec P'$; $extractFrame P = \langle A_P, \Psi_P \rangle$; $distinct A_P$;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \text{i}M\langle N \rangle \prec Q'$; $extractFrame Q = \langle A_Q, \Psi_Q \rangle$;
 $distinct A_Q$;
 $A_P \#^* \Psi; A_P \#^* \Psi_Q; A_P \#^* P; A_P \#^* N$;
 $A_P \#^* P'; A_P \#^* Q; A_P \#^* Q'; A_P \#^* M; A_P \#^* A_Q; A_P \#^* xvec$;
 $A_Q \#^* \Psi; A_Q \#^* \Psi_P$;
 $A_Q \#^* P; A_Q \#^* N; A_Q \#^* P'; A_Q \#^* M; A_Q \#^* Q; A_Q \#^* Q'; A_Q$
 $\#^* xvec$;
 $xvec \#^* \Psi; xvec \#^* \Psi_P; xvec \#^* \Psi_Q; xvec \#^* P; xvec \#^* Q; xvec \#^*$
 M ;
 $A_P \#^* C; A_Q \#^* C; xvec \#^* C; distinct xvec \rrbracket \implies Prop$
and $rBrClose$: $\bigwedge P M xvec N P' x$.
 $\llbracket cP = (\nu x)P; cRs = \tau \prec (\nu x)((\nu^*xvec)P') \rrbracket$;
 $x \in supp M$;
 $\Psi \triangleright P \mapsto \text{i}M(\nu^*xvec)\langle N \rangle \prec P'$;
 $distinct xvec; xvec \#^* \Psi; xvec \#^* P$;
 $xvec \#^* M$;
 $x \# \Psi; x \# xvec$;
 $xvec \#^* C; x \# C \rrbracket \implies Prop$
and $rOpen$: $\bigwedge P M xvec yvec N P' x$.
 $\llbracket cP = (\nu x)P; cRs = M(\nu^*(xvec @ x \# yvec))\langle N \rangle \prec P' \rrbracket$;
 $\Psi \triangleright P \mapsto M(\nu^*(xvec @ yvec))\langle N \rangle \prec P'$; $x \in supp N$; $x \# xvec$; $x \#$
 $yvec$; $x \# M$; $x \# \Psi$; $distinct xvec$; $distinct yvec$;
 $xvec \#^* \Psi; xvec \#^* P; xvec \#^* M; xvec \#^* yvec; yvec \#^* \Psi; yvec \#^* P$;
 $yvec \#^* M; xvec \#^* C; x \# C; yvec \#^* C \rrbracket \implies$
 $Prop$
and $rBrOpen$: $\bigwedge P M xvec yvec N P' x$.
 $\llbracket cP = (\nu x)P; cRs = \text{i}M(\nu^*(xvec @ x \# yvec))\langle N \rangle \prec P' \rrbracket$;
 $\Psi \triangleright P \mapsto \text{i}M(\nu^*(xvec @ yvec))\langle N \rangle \prec P'$; $x \in supp N$; $x \# xvec$; $x \#$
 $yvec$; $x \# M$; $x \# \Psi$; $distinct xvec$; $distinct yvec$;
 $xvec \#^* \Psi; xvec \#^* P; xvec \#^* M; xvec \#^* yvec; yvec \#^* \Psi; yvec \#^* P$;
 $yvec \#^* M; xvec \#^* C; x \# C; yvec \#^* C \rrbracket \implies$
 $Prop$
and $rScope$: $\bigwedge P \alpha P' x$. $\llbracket cP = (\nu x)P; cRs = \alpha \prec (\nu x)P' \rrbracket$;
 $\Psi \triangleright P \mapsto \alpha \prec P'$; $x \# \Psi$; $x \# \alpha$; $x \# C$; $bn \alpha \#^* \Psi$; $bn \alpha$
 $\#^* P$; $bn \alpha \#^* subject \alpha$; $bn \alpha \#^* C$; $distinct(bn \alpha) \rrbracket \implies Prop$
and $rBang$: $\bigwedge P$. $\llbracket cP = !P; \Psi \triangleright P \parallel !P \mapsto cRs; guarded P \rrbracket \implies Prop$
shows $Prop$
 $\langle proof \rangle$

nominal-primrec

$inputLength :: ('a::fs-name, 'b::fs-name, 'c::fs-name) psi \Rightarrow nat$
and $inputLength' :: ('a::fs-name, 'b::fs-name, 'c::fs-name) input \Rightarrow nat$
and $inputLength'' :: ('a::fs-name, 'b::fs-name, 'c::fs-name) psiCase \Rightarrow nat$

where

$inputLength \mathbf{0} = 0$
 $| inputLength (M(N).P) = 0$
 $| inputLength (M(I) = inputLength' I$
 $| inputLength (Case C) = 0$
 $| inputLength (P \parallel Q) = 0$
 $| inputLength (\nu x)P = 0$
 $| inputLength (\Psi) = 0$
 $| inputLength (!P) = 0$

$| inputLength' (Trm M P) = 0$
 $| inputLength' (\nu y I) = 1 + (inputLength' I)$

$| inputLength'' (\perp_c) = 0$
 $| inputLength'' (\Box \Phi \Rightarrow P C) = 0$
 $\quad \langle proof \rangle$

nominal-primrec $boundOutputLength :: ('a, 'b, 'c) boundOutput \Rightarrow nat$
where
 $boundOutputLength (BOut M P) = 0$
 $| boundOutputLength (BStep x B) = (boundOutputLength B) + 1$
 $\quad \langle proof \rangle$

nominal-primrec $residualLength :: ('a, 'b, 'c) residual \Rightarrow nat$
where
 $residualLength (RIn M N P) = 0$
 $| residualLength (RBrIn M N P) = 0$
 $| residualLength (ROut M B) = boundOutputLength B$
 $| residualLength (RBrOut M B) = boundOutputLength B$
 $| residualLength (RTau P) = 0$
 $\langle proof \rangle$

lemma $inputLengthProc[simp]$:
shows $inputLength(M(\lambda *xvec N).P) = length\ xvec$
 $\langle proof \rangle$

lemma $boundOutputLengthSimp[simp]$:
shows $residualLength(M(\nu *xvec)\langle N \rangle \prec P) = length\ xvec$
and $residualLength(\downarrow M(\nu *xvec)\langle N \rangle \prec P) = length\ xvec$
 $\langle proof \rangle$

lemma $boundOutputLengthSimp2[simp]$:
shows $residualLength(\alpha \prec P) = length(bn\ \alpha)$
 $\langle proof \rangle$

lemmas $[simp\ del] = inputLength-inputLength'-inputLength''.simps\ residualLength.simps\ boundOutputLength.simps$

lemma $constructPerm$:
fixes $xvec :: name\ list$

and $yvec :: name\ list$

assumes $length\ xvec = length\ yvec$
and $xvec \#* yvec$
and $distinct\ xvec$
and $distinct\ yvec$

obtains p **where** $set\ p \subseteq set\ xvec \times set(p \cdot xvec)$ **and** $distinctPerm\ p$ **and** $yvec = p \cdot xvec$
 $\langle proof \rangle$

lemma $distinctApend[simp]$:
fixes $xvec :: name\ list$
and $yvec :: name\ list$

shows $(set\ xvec \cap set\ yvec = \{\}) = xvec \#* yvec$
 $\langle proof \rangle$

lemma $lengthAux$:
fixes $xvec :: name\ list$
and $y :: name$
and $yvec :: name\ list$

assumes $length\ xvec = length(y\#yvec)$

obtains $z\ zvec$ **where** $xvec = z\#zvec$ **and** $length\ zvec = length\ yvec$
 $\langle proof \rangle$

lemma $lengthAux2$:
fixes $xvec :: name\ list$
and $yvec :: name\ list$
and $zvec :: name\ list$

assumes $length\ xvec = length(yvec@y\#zvec)$

obtains $xvec1\ x\ xvec2$ **where** $xvec = xvec1@x\#xvec2$ **and** $length\ xvec1 = length\ yvec$ **and** $length\ xvec2 = length\ zvec$
 $\langle proof \rangle$

lemma $semanticsCases[consumes\ 19, case-names\ cInput\ cBrInput\ cOutput\ cBrOutput\ cCase\ cPar1\ cPar2\ cComm1\ cComm2\ cBrMerge\ cBrComm1\ cBrComm2\ cBrClose\ cOpen\ cBrOpen\ cScope\ cBang]$:
fixes $\Psi :: 'b$
and $cP :: ('a, 'b, 'c)\ psi$
and $cRs :: ('a, 'b, 'c)\ residual$
and $C :: 'f::fs-name$
and $x1 :: name$
and $x2 :: name$
and $x3 :: name$

and $x_4 :: \text{name}$
and $xvec1 :: \text{name list}$
and $xvec2 :: \text{name list}$
and $xvec3 :: \text{name list}$
and $xvec_4 :: \text{name list}$
and $xvec5 :: \text{name list}$
and $xvec6 :: \text{name list}$
and $xvec7 :: \text{name list}$
and $xvec8 :: \text{name list}$
and $xvec9 :: \text{name list}$

assumes $\Psi \triangleright cP \mapsto cRs$

and $\text{length } xvec1 = \text{inputLength } cP$ **and** $\text{distinct } xvec1$
and $\text{length } xvec6 = \text{inputLength } cP$ **and** $\text{distinct } xvec6$
and $\text{length } xvec2 = \text{residualLength } cRs$ **and** $\text{distinct } xvec2$
and $\text{length } xvec3 = \text{residualLength } cRs$ **and** $\text{distinct } xvec3$
and $\text{length } xvec_4 = \text{residualLength } cRs$ **and** $\text{distinct } xvec_4$
and $\text{length } xvec5 = \text{residualLength } cRs$ **and** $\text{distinct } xvec5$
and $\text{length } xvec7 = \text{residualLength } cRs$ **and** $\text{distinct } xvec7$
and $\text{length } xvec8 = \text{residualLength } cRs$ **and** $\text{distinct } xvec8$
and $\text{length } xvec9 = \text{residualLength } cRs$ **and** $\text{distinct } xvec9$
and $rInput: \bigwedge M K N Tvec P. (\llbracket xvec1 \#* \Psi; xvec1 \#* cP; xvec1 \#* cRs \rrbracket \implies$
 $cP = M(\lambda * xvec1 N).P \wedge cRs = K(\langle N[xvec1 ::= Tvec] \rangle) \prec P[xvec1 ::= Tvec] \wedge$
 $\Psi \vdash M \leftrightarrow K \wedge \text{distinct } xvec1 \wedge \text{set } xvec1 \subseteq$
 $\text{supp } N \wedge \text{length } xvec1 = \text{length } Tvec \wedge$
 $xvec1 \#* Tvec \wedge xvec1 \#* \Psi \wedge xvec1 \#* M \wedge$
 $xvec1 \#* K) \implies Prop$
and $rBrInput: \bigwedge M K N Tvec P. (\llbracket xvec6 \#* \Psi; xvec6 \#* cP; xvec6 \#* cRs \rrbracket \implies$
 $cP = M(\lambda * xvec6 N).P \wedge cRs = \imath K(\langle N[xvec6 ::= Tvec] \rangle) \prec P[xvec6 ::= Tvec] \wedge$
 $\Psi \vdash K \succeq M \wedge \text{distinct } xvec6 \wedge \text{set } xvec6 \subseteq$
 $\text{supp } N \wedge \text{length } xvec6 = \text{length } Tvec \wedge$
 $xvec6 \#* Tvec \wedge xvec6 \#* \Psi \wedge xvec6 \#* M \wedge$
 $xvec6 \#* K) \implies Prop$
and $rOutput: \bigwedge M K N P. \llbracket cP = M\langle N \rangle.P; cRs = K\langle N \rangle \prec P; \Psi \vdash M \leftrightarrow K \rrbracket$
 $\implies Prop$
and $rBrOutput: \bigwedge M K N P. \llbracket cP = M\langle N \rangle.P; cRs = \imath K\langle N \rangle \prec P; \Psi \vdash M \preceq$
 $K \rrbracket \implies Prop$
and $rCase: \bigwedge Cs P \varphi. \llbracket cP = \text{Cases } Cs; \Psi \triangleright P \mapsto cRs; (\varphi, P) \in \text{set } Cs; \Psi \vdash$
 $\varphi; \text{guarded } P \rrbracket \implies Prop$
and $rPar1: \bigwedge \Psi_Q P \alpha P' Q A_Q. (\llbracket xvec2 \#* \Psi; xvec2 \#* cP; xvec2 \#* cRs \rrbracket \implies$
 $cP = P \parallel Q \wedge cRs = \alpha \prec (P' \parallel Q) \wedge xvec2 =$
 $bn \alpha \wedge$
 $\Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P' \wedge \text{extractFrame } Q =$
 $\langle A_Q, \Psi_Q \rangle \wedge \text{distinct } A_Q \wedge$
 $A_Q \#* P \wedge A_Q \#* Q \wedge A_Q \#* \Psi \wedge A_Q \#* \alpha \wedge$
 $A_Q \#* P' \wedge A_Q \#* C) \implies Prop$
and $rPar2: \bigwedge \Psi_P Q \alpha Q' P A_P. (\llbracket xvec3 \#* \Psi; xvec3 \#* cP; xvec3 \#* cRs \rrbracket \implies$
 $cP = P \parallel Q \wedge cRs = \alpha \prec (P \parallel Q') \wedge xvec3 =$
 $bn \alpha \wedge$

$\Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q' \wedge \text{extractFrame } P =$

$\langle A_P, \Psi_P \rangle \wedge \text{distinct } A_P \wedge$

$A_P \#* P \wedge A_P \#* Q \wedge A_P \#* \Psi \wedge A_P \#* \alpha \wedge$

$A_P \#* Q' \wedge A_P \#* C \implies \text{Prop}$

and $rComm1: \bigwedge \Psi_Q P M N P' A_P \Psi_P Q K \text{vec } Q' A_Q.$

$\llbracket cP = P \parallel Q; cRs = \tau \prec (\nu * \text{vec}) P' \parallel Q' \rrbracket;$

$\Psi \otimes \Psi_Q \triangleright P \mapsto M(N) \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$

$\text{distinct } A_P;$

$\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * \text{vec})(N) \prec Q'; \text{extractFrame } Q = \langle A_Q,$

$\Psi_Q \rangle; \text{distinct } A_Q;$

$\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#*$

$M; A_P \#* N;$

$A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{vec}; A_Q \#* \Psi;$

$A_Q \#* \Psi_P;$

$A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; A_Q$

$\#* \text{vec};$

$\text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* \Psi_Q; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#*$

$Q;$

$\text{vec} \#* K; A_P \#* C; A_Q \#* C; \text{vec} \#* C; \text{distinct } \text{vec} \rrbracket \implies \text{Prop}$

and $rComm2: \bigwedge \Psi_Q P M \text{vec } N P' A_P \Psi_P Q K Q' A_Q.$

$\llbracket cP = P \parallel Q; cRs = \tau \prec (\nu * \text{vec}) P' \parallel Q' \rrbracket;$

$\Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * \text{vec})(N) \prec P'; \text{extractFrame } P = \langle A_P,$

$\Psi_P \rangle; \text{distinct } A_P;$

$\Psi \otimes \Psi_P \triangleright Q \mapsto K(N) \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$

$\text{distinct } A_Q;$

$\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#*$

$M; A_P \#* N;$

$A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* \text{vec}; A_Q \#* \Psi;$

$A_Q \#* \Psi_P;$

$A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; A_Q$

$\#* \text{vec};$

$\text{vec} \#* \Psi; \text{vec} \#* \Psi_P; \text{vec} \#* \Psi_Q; \text{vec} \#* P; \text{vec} \#* M; \text{vec} \#*$

$Q;$

$\text{vec} \#* K; A_P \#* C; A_Q \#* C; \text{vec} \#* C; \text{distinct } \text{vec} \rrbracket \implies \text{Prop}$

and $rBrMerge: \bigwedge \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q.$

$\llbracket cP = (P \parallel Q); cRs = \imath M(N) \prec (P' \parallel Q') \rrbracket;$

$\Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(N) \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$

$\text{distinct } A_P;$

$\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(N) \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$

$\text{distinct } A_Q;$

$A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$

$A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$

$A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$

$A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C \rrbracket \implies \text{Prop}$

and $rBrComm1: \bigwedge \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q.$

$\llbracket \text{vec} \#* \Psi; \text{vec} \#* cP; \text{vec} \#* cRs \rrbracket \implies$

$cP = P \parallel Q \wedge cRs = \imath M(\nu * \text{vec} \#*) (N) \prec (P' \parallel Q') \wedge$

$\Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(N) \prec P' \wedge \text{extractFrame } P = \langle A_P, \Psi_P \rangle \wedge$

$\text{distinct } A_P \wedge$

$$\begin{aligned}
& \Psi \otimes \Psi_P \triangleright Q \mapsto \mathfrak{!}M(\nu^*xvec7)\langle N \rangle \prec Q' \wedge extractFrame Q = \\
& \langle A_Q, \Psi_Q \rangle \wedge distinct A_Q \wedge \\
& \quad A_P \#* \Psi \wedge A_P \#* \Psi_Q \wedge A_P \#* P \wedge A_P \#* N \wedge \\
& \quad A_P \#* P' \wedge A_P \#* Q \wedge A_P \#* Q' \wedge A_P \#* A_Q \wedge A_P \#* xvec7 \wedge \\
& A_Q \#* \Psi \wedge A_Q \#* \Psi_P \wedge \\
& \quad A_Q \#* P \wedge A_Q \#* N \wedge A_Q \#* P' \wedge A_Q \#* Q \wedge A_Q \#* Q' \wedge A_Q \#* \\
& xvec7 \wedge \\
& \quad xvec7 \#* \Psi \wedge xvec7 \#* \Psi_P \wedge xvec7 \#* \Psi_Q \wedge xvec7 \#* P \wedge xvec7 \\
& \#* Q \wedge \\
& \quad A_P \#* M \wedge A_Q \#* M \wedge xvec7 \#* M \wedge \\
& \quad A_P \#* C \wedge A_Q \#* C \wedge distinct xvec7) \implies Prop \\
\text{and } rBrComm2: & \bigwedge \Psi_Q P M N P' A_P \Psi_P Q Q' A_Q. \\
& ([xvec8 \#* \Psi; xvec8 \#* cP; xvec8 \#* cRs] \implies \\
& \quad cP = P \parallel Q \wedge cRs = \mathfrak{!}M(\nu^*xvec8)\langle N \rangle \prec (P' \parallel Q') \wedge \\
& \Psi \otimes \Psi_Q \triangleright P \mapsto \mathfrak{!}M(\nu^*xvec8)\langle N \rangle \prec P' \wedge extractFrame P = \langle A_P, \\
\Psi_P \rangle \wedge distinct A_P \wedge \\
& \Psi \otimes \Psi_P \triangleright Q \mapsto \mathfrak{!}M\langle N \rangle \prec Q' \wedge extractFrame Q = \langle A_Q, \Psi_Q \rangle \wedge \\
distinct A_Q \wedge \\
& \quad A_P \#* \Psi \wedge A_P \#* \Psi_Q \wedge A_P \#* P \wedge A_P \#* N \wedge \\
& \quad A_P \#* P' \wedge A_P \#* Q \wedge A_P \#* Q' \wedge A_P \#* A_Q \wedge A_P \#* xvec8 \wedge \\
& A_Q \#* \Psi \wedge A_Q \#* \Psi_P \wedge \\
& \quad A_Q \#* P \wedge A_Q \#* N \wedge A_Q \#* P' \wedge A_Q \#* Q \wedge A_Q \#* Q' \wedge A_Q \#* \\
& xvec8 \wedge \\
& \quad xvec8 \#* \Psi \wedge xvec8 \#* \Psi_P \wedge xvec8 \#* \Psi_Q \wedge xvec8 \#* P \wedge xvec8 \\
& \#* Q \wedge \\
& \quad A_P \#* M \wedge A_Q \#* M \wedge xvec8 \#* M \wedge \\
& \quad A_P \#* C \wedge A_Q \#* C \wedge distinct xvec8) \implies Prop \\
\text{and } rBrClose: & \bigwedge P M N xvec P'. \\
& ([x3 \#* \Psi; x3 \#* cP; x3 \#* cRs] \implies \\
& \quad cP = (\nu x3)P \wedge cRs = \tau \prec (\nu x3)((\nu^*xvec)P') \wedge \\
& \quad x3 \in supp M \wedge \\
& \quad \Psi \triangleright P \mapsto \mathfrak{!}M(\nu^*xvec)\langle N \rangle \prec P' \wedge \\
& \quad distinct xvec \wedge xvec \#* \Psi \wedge xvec \#* P \wedge \\
& \quad xvec \#* M \wedge xvec \#* C \wedge \\
& \quad x3 \#* \Psi \wedge x3 \#* xvec) \implies Prop \\
\text{and } rOpen: & \bigwedge P M xvec y yvec N P'. \\
& ([xvec4 \#* \Psi; xvec4 \#* cP; xvec4 \#* cRs; x1 \#* \Psi; x1 \#* cP; x1 \#* \\
& cRs; x1 \#* xvec4] \implies \\
& \quad cP = (\nu x1)P \wedge cRs = M(\nu^*(xvec@x1\#yvec))\langle N \rangle \prec P' \wedge \\
& xvec4 = xvec@y\#yvec \wedge \\
& \quad \Psi \triangleright P \mapsto M(\nu^*(xvec@yvec))\langle N \rangle \prec P' \wedge x1 \in supp N \wedge x1 \#* \\
& xvec \wedge x1 \#* yvec \wedge \\
& \quad distinct xvec \wedge distinct yvec \wedge xvec \#* \Psi \wedge xvec \#* P \wedge xvec \#* M \\
& \wedge xvec \#* yvec \wedge \\
& \quad yvec \#* \Psi \wedge yvec \#* P \wedge yvec \#* M) \implies Prop \\
\text{and } rBrOpen: & \bigwedge P M xvec y yvec N P'. \\
& ([xvec9 \#* \Psi; xvec9 \#* cP; xvec9 \#* cRs; x4 \#* \Psi; x4 \#* cP; x4 \#* \\
& cRs; x4 \#* xvec9] \implies \\
& \quad cP = (\nu x4)P \wedge cRs = \mathfrak{!}M(\nu^*(xvec@x4\#yvec))\langle N \rangle \prec P' \wedge
\end{aligned}$$

$xvec9 = xvec @ y \# yvec \wedge$
 $\Psi \triangleright P \mapsto \text{iM}(\nu^*(xvec @ yvec)) \langle N \rangle \prec P' \wedge x4 \in \text{supp } N \wedge x4 \#$
 $xvec \wedge x4 \# yvec \wedge$
 $\text{distinct } xvec \wedge \text{distinct } yvec \wedge xvec \# \Psi \wedge xvec \# P \wedge xvec \# M$
 $\wedge xvec \# yvec \wedge$
 $yvec \# \Psi \wedge yvec \# P \wedge yvec \# M \implies \text{Prop}$
and $rScope: \bigwedge P \alpha P'. (\llbracket xvec5 \# \Psi; xvec5 \# cP; xvec5 \# cRs; x2 \# \Psi; x2 \#$
 $cP; x2 \# cRs; x2 \# xvec5 \rrbracket \implies$
 $cP = (\nu x2)P \wedge cRs = \alpha \prec (\nu x2)P' \wedge xvec5 = \text{bn } \alpha \wedge$
 $\Psi \triangleright P \mapsto \alpha \prec P' \wedge x2 \# \Psi \wedge x2 \# \alpha \wedge \text{bn } \alpha \# \text{subject}$
 $\alpha \wedge \text{distinct}(\text{bn } \alpha) \implies \text{Prop}$
and $rBang: \bigwedge P. \llbracket cP = !P; \Psi \triangleright P \parallel !P \mapsto cRs; \text{guarded } P \rrbracket \implies \text{Prop}$
shows Prop
 $\langle \text{proof} \rangle$

lemma resResidEq :

fixes $xvec :: \text{name list}$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$
and $M :: 'a$
and $N :: 'a$
and $N' :: 'a$

assumes $\text{iM}(\nu^*xvec) \langle N \rangle \prec P = \text{iM}(\nu^*xvec) \langle N' \rangle \prec Q$
and $xvec \# M$

shows $\text{iM}(\nu^*xvec) \langle N \rangle = \text{iM}(\nu^*xvec) \langle N' \rangle$
 $\langle \text{proof} \rangle$

lemma parCases [*consumes 5, case-names cPar1 cPar2 cComm1 cComm2 cBrMerge cBrComm1 cBrComm2*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$
and $\alpha :: 'a \text{ action}$
and $T :: ('a, 'b, 'c) \text{psi}$
and $C :: 'f :: \text{fs-name}$
and $M :: 'a$
and $N :: 'a$

assumes $\text{Trans}: \Psi \triangleright P \parallel Q \mapsto \alpha \prec T$

and $\text{bn } \alpha \# \Psi$
and $\text{bn } \alpha \# P$
and $\text{bn } \alpha \# Q$
and $\text{bn } \alpha \# \text{subject } \alpha$
and $rPar1: \bigwedge P' A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P'; \text{extractFrame } Q = \langle A_Q,$
 $\Psi_Q \rangle; \text{distinct } A_Q;$

$A_Q \# \Psi; A_Q \# P; A_Q \# Q; A_Q \# \alpha; A_Q \# P'; A_Q$

$\#* C \Longrightarrow Prop \alpha (P' \parallel Q)$
and $rPar2: \bigwedge Q' A_P \Psi_P. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q'; extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* \alpha; A_P \#* Q'; A_P \#*$
 $C \Longrightarrow Prop \alpha (P \parallel Q')$
and $rComm1: \bigwedge \Psi_Q M N P' A_P \Psi_P K xvec Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M \langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle; distinct A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K \langle \nu * xvec \rangle \langle N \rangle \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; distinct xvec; \alpha = \tau;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P$
 $\#* Q; A_P \#* xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#*$
 $Q; A_Q \#* xvec; A_Q \#* Q'; A_Q \#* C;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* M; xvec \#* K; xvec \#* Q;$
 $xvec \#* \Psi_Q; xvec \#* C \Longrightarrow$
 $Prop (\tau) (\langle \nu * xvec \rangle (P' \parallel Q'))$
and $rComm2: \bigwedge \Psi_Q M xvec N P' A_P \Psi_P K Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M \langle \nu * xvec \rangle \langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
 $distinct A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K \langle N \rangle \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle; distinct A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; distinct xvec; \alpha = \tau;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P$
 $\#* Q; A_P \#* xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#*$
 $Q; A_Q \#* xvec; A_Q \#* Q'; A_Q \#* C;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* M; xvec \#* K; xvec \#* Q;$
 $xvec \#* \Psi_Q; xvec \#* C \Longrightarrow$
 $Prop (\tau) (\langle \nu * xvec \rangle (P' \parallel Q'))$
and $rBrMerge: \bigwedge \Psi_Q M N P' A_P \Psi_P Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \iota M \langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$
 $distinct A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \iota M \langle N \rangle \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C; \alpha = \iota M \langle N \rangle \Longrightarrow$
 $Prop (\iota M \langle N \rangle) (P' \parallel Q')$
and $rBrComm1: \bigwedge \Psi_Q M N P' A_P \Psi_P xvec Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \iota M \langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle; distinct$
 $A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \iota M \langle \nu * xvec \rangle \langle N \rangle \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$
 $distinct xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#*$
 $xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#*$
 $xvec; A_Q \#* Q'; A_Q \#* C;$

$A_P \#* M; A_Q \#* M; xvec \#* M; \text{iM}(\nu*xvec)\langle N \rangle = \alpha;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* Q; xvec \#* \Psi_Q \implies$
 $\text{Prop} (\text{iM}(\nu*xvec)\langle N \rangle) (P' \parallel Q')$
and $rBrComm2: \bigwedge \Psi_Q M xvec N P' A_P \Psi_P Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \text{iM}(\nu*xvec)\langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
distinct $A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \text{iM}(\nu*xvec)\langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$ *distinct*
 $A_Q;$
distinct $xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#*$
 $xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#*$
 $xvec; A_Q \#* Q'; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M; \text{iM}(\nu*xvec)\langle N \rangle = \alpha;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* Q; xvec \#* \Psi_Q \implies$
 $\text{Prop} (\text{iM}(\nu*xvec)\langle N \rangle) (P' \parallel Q')$

shows $\text{Prop } \alpha T$
 $\langle \text{proof} \rangle$

lemma $\text{parInputCases}[\text{consumes } 1, \text{case-names } cPar1 \ cPar2]:$

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$
and $M :: 'a$
and $N :: 'a$
and $R :: ('a, 'b, 'c) \text{psi}$
and $C :: 'f::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright P \parallel Q \mapsto M(\nu N) \prec R$

and $rPar1: \bigwedge P' A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu N) \prec P'; \text{extractFrame } Q =$
 $\langle A_Q, \Psi_Q \rangle;$ *distinct* $A_Q;$

$A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M; A_Q \#* N; A_Q \#* C \implies$
 $\text{Prop} (P' \parallel Q)$

and $rPar2: \bigwedge Q' A_P \Psi_P. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto M(\nu N) \prec Q'; \text{extractFrame } P =$
 $\langle A_P, \Psi_P \rangle;$ *distinct* $A_P;$

$A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* M; A_P \#* N; A_P \#* C \implies$
 $\text{Prop} (P \parallel Q')$

shows $\text{Prop } R$
 $\langle \text{proof} \rangle$

lemma $\text{parBrInputCases}[\text{consumes } 1, \text{case-names } cPar1 \ cPar2 \ cBrMerge]:$

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$
and $M :: 'a$
and $N :: 'a$
and $R :: ('a, 'b, 'c) \text{psi}$
and $C :: 'f::\text{fs-name}$

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto_{iM} \langle N \rangle \prec R$
and *rPar1*: $\bigwedge P' A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_{iM} \langle N \rangle \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M; A_Q \#* N; A_Q \#* C \rrbracket$
 $\implies \text{Prop } (P' \parallel Q)$
and *rPar2*: $\bigwedge Q' A_P \Psi_P. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_{iM} \langle N \rangle \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* M; A_P \#* N; A_P \#* C \rrbracket$
 $\implies \text{Prop } (P \parallel Q')$
and *rBrMerge*: $\bigwedge \Psi_Q P' A_P \Psi_P Q' A_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_{iM} \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; \Psi \otimes \Psi_P \triangleright Q \mapsto_{iM} \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M; A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C \rrbracket \implies \text{Prop } (P' \parallel Q')$
shows *Prop R*
 $\langle \text{proof} \rangle$

lemma *parOutputCases*[*consumes 5, case-names cPar1 cPar2*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$
and $M :: 'a$
and $xvec :: \text{name list}$
and $N :: 'a$
and $R :: ('a, 'b, 'c) \text{psi}$
and $C :: 'f::\text{fs-name}$

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto_{M(\nu * xvec)} \langle N \rangle \prec R$
and $xvec \#* \Psi$
and $xvec \#* P$
and $xvec \#* Q$
and $xvec \#* M$
and *rPar1*: $\bigwedge P' A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_{M(\nu * xvec)} \langle N \rangle \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M; A_Q \#* xvec; A_Q \#* N; A_Q \#* C; A_Q \#* xvec; \text{distinct } xvec \rrbracket \implies \text{Prop } (P' \parallel Q)$
and *rPar2*: $\bigwedge Q' A_P \Psi_P. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_{M(\nu * xvec)} \langle N \rangle \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P; A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* M; A_P \#* xvec; A_P \#* N; A_P \#* C; A_P \#* xvec; \text{distinct } xvec \rrbracket \implies \text{Prop } (P \parallel Q')$
shows *Prop R*
 $\langle \text{proof} \rangle$

lemma *parBrOutputCases*[*consumes 5, case-names cPar1 cPar2 cBrComm1 cBrComm2*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $xvec :: \text{name list}$
and $N :: 'a$
and $R :: ('a, 'b, 'c) \text{ psi}$
and $C :: 'f::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright P \parallel Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec R$
and $xvec \#* \Psi$
and $xvec \#* P$
and $xvec \#* Q$
and $xvec \#* M$
and $rPar1: \bigwedge P' A_Q \Psi_Q. [\Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M; A_Q \#* xvec; A_Q \#* N;$
 $A_Q \#* C; A_Q \#* xvec; \text{distinct } xvec] \implies \text{Prop } (P' \parallel Q)$
and $rPar2: \bigwedge Q' A_P \Psi_P. [\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* M; A_P \#* xvec; A_P \#* N;$
 $A_P \#* C; A_P \#* xvec; \text{distinct } xvec] \implies \text{Prop } (P \parallel Q')$
and $rBrComm1: \bigwedge \Psi_Q P' A_P \Psi_P Q' A_Q.$
 $[\Psi \otimes \Psi_Q \triangleright P \mapsto_i M \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu * xvec) \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
 $\text{distinct } A_Q;$
 $\text{distinct } xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#*$
 $xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#*$
 $xvec; A_Q \#* Q'; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* Q; xvec \#* \Psi_Q] \implies$
 $\text{Prop } (P' \parallel Q')$
and $rBrComm2: \bigwedge \Psi_Q P' A_P \Psi_P Q' A_Q.$
 $[\Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu * xvec) \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
 $\text{distinct } A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{distinct } A_Q;$
 $\text{distinct } xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#*$
 $xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#*$
 $xvec; A_Q \#* Q'; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* Q; xvec \#* \Psi_Q] \implies$

```

      Prop (P' || Q')
shows Prop R
  <proof>

lemma theEqvt[eqt-force]:
  fixes p :: name prm
    and α :: 'a action

assumes α ≠ τ

shows (p · the(subject α)) = the(p · (subject α))
  <proof>

lemma theSubjectFresh[simp]:
  fixes α :: 'a action
    and x :: name

assumes α ≠ τ

shows x # the(subject α) = x # subject α
  <proof>

lemma theSubjectFreshChain[simp]:
  fixes α :: 'a action
    and xvec :: name list

assumes α ≠ τ

shows xvec #* the(subject α) = xvec #* subject α
  <proof>

lemma inputObtainPrefix:
  fixes Ψ :: 'b
    and P :: ('a, 'b, 'c) psi
    and P' :: ('a, 'b, 'c) psi
    and AP :: name list
    and ΨP :: 'b
    and N :: 'a
    and K :: 'a
    and B :: name list

assumes Ψ ▷ P ⟶ K(N) < P'
  and extractFrame P = ⟨AP, ΨP⟩
  and distinct AP
  and B #* P
  and AP #* Ψ
  and AP #* B
  and AP #* P
  and AP #* K

```


obtains M where $\Psi \otimes \Psi_P \vdash K \leftrightarrow M$ and $B \#^* M$
<proof>

lemma *outputObtainPrefix*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $A_P :: \text{ name list}$
and $\Psi_P :: 'b$
and $N :: 'a$
and $K :: 'a$
and $xvec :: \text{ name list}$
and $B :: \text{ name list}$

assumes $\Psi \triangleright P \mapsto ROut\ K\ ((\nu^*xvec)N \prec' P')$

and $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $xvec \#^* K$
and $distinct\ xvec$
and $B \#^* P$
and $A_P \#^* \Psi$
and $A_P \#^* B$
and $A_P \#^* P$
and $A_P \#^* K$

obtains M where $\Psi \otimes \Psi_P \vdash K \leftrightarrow M$ and $B \#^* M$
<proof>

lemma *inputRenameSubject*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $A_P :: \text{ name list}$
and $\Psi_P :: 'b$

assumes $\Psi \triangleright P \mapsto M(N) \prec P'$

and $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $\Psi \otimes \Psi_P \vdash M \leftrightarrow K$
and $A_P \#^* \Psi$
and $A_P \#^* P$
and $A_P \#^* M$
and $A_P \#^* K$

shows $\Psi \triangleright P \mapsto K(N) \prec P'$

<proof>

lemma *outputRenameSubject*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and A_P :: name list
and Ψ_P :: 'b

assumes $\Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$

and $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $\Psi \otimes \Psi_P \vdash M \leftrightarrow K$
and $A_P \#* \Psi$
and $A_P \#* P$
and $A_P \#* M$
and $A_P \#* K$

shows $\Psi \triangleright P \mapsto K(\nu*xvec)\langle N \rangle \prec P'$

<proof>

lemma *parCasesSubject*[*consumes* 7, *case-names* $cPar1\ cPar2\ cComm1\ cComm2$ $cBrMerge\ cBrComm1\ cBrComm2$]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and Q :: ('a, 'b, 'c) psi
and α :: 'a action
and R :: ('a, 'b, 'c) psi
and C :: 'f::fs-name
and $yvec$:: name list

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto \alpha \prec R$

and $bn\ \alpha \#* \Psi$
and $bn\ \alpha \#* P$
and $bn\ \alpha \#* Q$
and $bn\ \alpha \#* subject\ \alpha$
and $yvec \#* P$
and $yvec \#* Q$
and $rPar1$: $\bigwedge P' A_Q \Psi_Q. [\Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P';\ extractFrame\ Q = \langle A_Q,$
 $\Psi_Q \rangle;\ distinct\ A_Q;$
 $A_Q \#* \Psi;\ A_Q \#* P;\ A_Q \#* \alpha;\ A_Q \#* C] \implies Prop\ \alpha\ (P' \parallel Q)$
and $rPar2$: $\bigwedge Q' A_P \Psi_P. [\Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q';\ extractFrame\ P = \langle A_P,$
 $\Psi_P \rangle;\ distinct\ A_P;$
 $A_P \#* \Psi;\ A_P \#* Q;\ A_P \#* \alpha;\ A_P \#* C] \implies Prop\ \alpha\ (P \parallel Q')$
and $rComm1$: $\bigwedge \Psi_Q\ M\ N\ P'\ A_P\ \Psi_P\ K\ xvec\ Q'\ A_Q.$
 $[\Psi \otimes \Psi_Q \triangleright P \mapsto M\langle N \rangle \prec P';\ extractFrame\ P = \langle A_P, \Psi_P \rangle;\ distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu*xvec)\langle N \rangle \prec Q';\ extractFrame\ Q = \langle A_Q, \Psi_Q \rangle;$

$distinct\ A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; yvec \#* M; yvec \#* K; distinct\ xvec; \alpha = \tau;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* Q;$
 $A_P \#* xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#* Q;$
 $A_Q \#* xvec; A_Q \#* Q'; A_Q \#* C;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* M; xvec \#* K; xvec \#* Q;$
 $xvec \#* \Psi_Q; xvec \#* C \implies$
 $Prop(\tau)(\llbracket \nu * xvec \rrbracket (P' \parallel Q'))$
and $rComm2: \bigwedge \Psi_Q\ M\ xvec\ N\ P'\ A_P\ \Psi_P\ K\ Q'\ A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'; extractFrame\ P = \langle A_P, \Psi_P \rangle;$
 $distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto K(\llbracket N \rrbracket) \prec Q'; extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$
 $\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; yvec \#* M; yvec \#* K; distinct\ xvec; \alpha = \tau;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* M; A_P \#* N; A_P \#* P'; A_P \#* Q;$
 $A_P \#* xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* K; A_Q \#* N; A_Q \#* P'; A_Q \#* Q;$
 $A_Q \#* xvec; A_Q \#* Q'; A_Q \#* C;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* M; xvec \#* K; xvec \#* Q;$
 $xvec \#* \Psi_Q; xvec \#* C \implies$
 $Prop(\tau)(\llbracket \nu * xvec \rrbracket (P' \parallel Q'))$
and $rBrMerge: \bigwedge \Psi_Q\ M\ N\ P'\ A_P\ \Psi_P\ Q'\ A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(\llbracket N \rrbracket) \prec P'; extractFrame\ P = \langle A_P, \Psi_P \rangle;$
 $distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\llbracket N \rrbracket) \prec Q'; extractFrame\ Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct\ A_Q;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P';$
 $A_P \#* Q; A_P \#* Q'; A_P \#* A_Q; A_P \#* M; A_Q \#* M;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P';$
 $A_Q \#* Q; A_Q \#* Q'; A_P \#* C; A_Q \#* C; \alpha = \imath M(\llbracket N \rrbracket) \implies$
 $Prop(\imath M(\llbracket N \rrbracket))(P' \parallel Q')$
and $rBrComm1: \bigwedge \Psi_Q\ M\ N\ P'\ A_P\ \Psi_P\ xvec\ Q'\ A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(\llbracket N \rrbracket) \prec P'; extractFrame\ P = \langle A_P, \Psi_P \rangle; distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\nu * xvec)\langle N \rangle \prec Q'; extractFrame\ Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct\ A_Q;$
 $distinct\ xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#* xvec;$
 $A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#* xvec;$
 $A_Q \#* Q'; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M; \imath M(\nu * xvec)\langle N \rangle = \alpha;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* Q; xvec \#* \Psi_Q \implies$
 $Prop(\imath M(\nu * xvec)\langle N \rangle)(P' \parallel Q')$
and $rBrComm2: \bigwedge \Psi_Q\ M\ xvec\ N\ P'\ A_P\ \Psi_P\ Q'\ A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(\nu * xvec)\langle N \rangle \prec P'; extractFrame\ P = \langle A_P, \Psi_P \rangle;$
 $distinct\ A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\llbracket N \rrbracket) \prec Q'; extractFrame\ Q = \langle A_Q, \Psi_Q \rangle; distinct\ A_Q;$

$distinct\ xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* N; A_P \#* P'; A_P \#* Q; A_P \#*$
 $xvec; A_P \#* Q'; A_P \#* A_Q; A_P \#* C;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* N; A_Q \#* P'; A_Q \#* Q; A_Q \#*$
 $xvec; A_Q \#* Q'; A_Q \#* C;$
 $A_P \#* M; A_Q \#* M; xvec \#* M; \text{i}M(\nu*xvec)\langle N \rangle = \alpha;$
 $xvec \#* \Psi; xvec \#* \Psi_P; xvec \#* P; xvec \#* Q; xvec \#* \Psi_Q \implies$
 $Prop (\text{i}M(\nu*xvec)\langle N \rangle) (P' \parallel Q')$

shows $Prop\ \alpha\ R$
 $\langle proof \rangle$

lemma $inputCases[consumes\ 1, case-names\ cInput\ cBrInput]:$

fixes $\Psi :: 'b$
and $M :: 'a$
and $xvec :: name\ list$
and $N :: 'a$
and $P :: ('a, 'b, 'c)\ psi$
and $\alpha :: 'a\ action$
and $P' :: ('a, 'b, 'c)\ psi$

assumes $Trans: \Psi \triangleright M(\lambda*xvec\ N).P \longmapsto \alpha \prec P'$

and $rInput: \bigwedge K\ Tvec. \llbracket \Psi \vdash M \leftrightarrow K; set\ xvec \subseteq supp\ N; length\ xvec = length\ Tvec; distinct\ xvec \rrbracket \implies Prop (K(N[xvec::=Tvec])) (P[xvec::=Tvec])$
and $rBrInput: \bigwedge K\ Tvec. \llbracket \Psi \vdash K \succeq M; set\ xvec \subseteq supp\ N; length\ xvec = length\ Tvec; distinct\ xvec \rrbracket \implies Prop (\text{i}K(N[xvec::=Tvec])) (P[xvec::=Tvec])$

shows $Prop\ \alpha\ P'$
 $\langle proof \rangle$

lemma $outputCases[consumes\ 1, case-names\ cOutput\ cBrOutput]:$

fixes $\Psi :: 'b$
and $M :: 'a$
and $N :: 'a$
and $P :: ('a, 'b, 'c)\ psi$
and $\alpha :: 'a\ action$
and $P' :: ('a, 'b, 'c)\ psi$

assumes $\Psi \triangleright M(N).P \longmapsto \alpha \prec P'$

and $\bigwedge K. \Psi \vdash M \leftrightarrow K \implies Prop (K\langle N \rangle) P$
and $\bigwedge K. \Psi \vdash M \preceq K \implies Prop (\text{i}K\langle N \rangle) P$

shows $Prop\ \alpha\ P'$
 $\langle proof \rangle$

lemma $caseCases[consumes\ 1, case-names\ cCase]:$

fixes $\Psi :: 'b$
and $Cs :: ('c \times ('a, 'b, 'c)\ psi)\ list$
and $\alpha :: 'a\ action$

and $P' :: ('a, 'b, 'c) \text{ psi}$

assumes $\text{Trans}: \Psi \triangleright (\text{Cases } Cs) \mapsto Rs$
and $r\text{Case}: \bigwedge \varphi P. \llbracket \Psi \triangleright P \mapsto Rs; (\varphi, P) \in \text{set } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \Longrightarrow \text{Prop}$

shows Prop
 $\langle \text{proof} \rangle$

lemma $\text{resCases}[\text{consumes } 7, \text{case-names } c\text{Open } c\text{BrOpen } c\text{Res } c\text{BrClose}]$:
fixes $\Psi :: 'b$
and $x :: \text{name}$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $\alpha :: 'a \text{ action}$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $C :: 'f::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright (\nu x)P \mapsto \alpha \prec P'$
and $x \# \Psi$
and $x \# \alpha$
and $x \# P'$
and $bn \alpha \#* \Psi$
and $bn \alpha \#* P$
and $bn \alpha \#* \text{subject } \alpha$
and $r\text{Open}: \bigwedge M \text{ xvec } yvec \ y \ N \ P'. \llbracket \Psi \triangleright P \mapsto M(\nu*(\text{xvec}@yvec)) \langle \langle [(x, y)] \cdot N \rangle \rangle \prec \langle \langle [(x, y)] \cdot P' \rangle \rangle; y \in \text{supp } N;$
 $x \# N; x \# P'; x \neq y; y \# \text{xvec}; y \# yvec; y \# M;$
 $\text{distinct } \text{xvec}; \text{distinct } yvec;$
 $yvec \#* P; \text{xvec} \#* M; y \# M;$
 $yvec \#* M; \text{xvec} \#* yvec \rrbracket \Longrightarrow$
 $\text{Prop } (M(\nu*(\text{xvec}@y\#yvec)) \langle N \rangle) P'$
and $r\text{BrOpen}: \bigwedge M \text{ xvec } yvec \ y \ N \ P'. \llbracket \Psi \triangleright P \mapsto {}_i M(\nu*(\text{xvec}@yvec)) \langle \langle [(x, y)] \cdot N \rangle \rangle \prec \langle \langle [(x, y)] \cdot P' \rangle \rangle; y \in \text{supp } N;$
 $x \# N; x \# P'; x \neq y; y \# \text{xvec}; y \# yvec; y \# M;$
 $\text{distinct } \text{xvec}; \text{distinct } yvec;$
 $yvec \#* P; \text{xvec} \#* M; y \# M;$
 $yvec \#* M; \text{xvec} \#* yvec \rrbracket \Longrightarrow$
 $\text{Prop } ({}_i M(\nu*(\text{xvec}@y\#yvec)) \langle N \rangle) P'$
and $r\text{Scope}: \bigwedge P'. \llbracket \Psi \triangleright P \mapsto \alpha \prec P' \rrbracket \Longrightarrow \text{Prop } \alpha ((\nu x)P')$
and $r\text{BrClose}: \bigwedge M \text{ xvec } N \ P'.$
 $\llbracket \Psi \triangleright P \mapsto {}_i M(\nu*\text{xvec}) \langle N \rangle \prec P';$
 $x \in \text{supp } M;$
 $\text{distinct } \text{xvec}; \text{xvec} \#* \Psi; \text{xvec} \#* P;$
 $\text{xvec} \#* M;$
 $x \# \Psi; x \# \text{xvec};$
 $\text{xvec} \#* C \rrbracket \Longrightarrow \text{Prop } (\tau) ((\nu x)((\nu*\text{xvec})P'))$

shows $\text{Prop } \alpha P'$

<proof>

lemma *resCases'*[*consumes 7, case-names cOpen cBrOpen cRes cBrClose*]:

fixes Ψ :: 'b
and x :: name
and P :: ('a, 'b, 'c) psi
and α :: 'a action
and P' :: ('a, 'b, 'c) psi
and C :: 'f::fs-name

assumes *Trans*: $\Psi \triangleright (\nu x)P \mapsto \alpha \prec P'$

and $x \# \Psi$

and $x \# \alpha$

and $x \# P'$

and $bn \alpha \#* \Psi$

and $bn \alpha \#* P$

and $bn \alpha \#* \text{subject } \alpha$

and *rOpen*: $\bigwedge M \text{ xvec yvec } y N P'. \llbracket \Psi \triangleright ((x, y) \cdot P) \mapsto M(\nu*(\text{xvec}@yvec)) \langle N \rangle \prec P'; y \in \text{supp } N;$

$x \# N; x \# P'; x \neq y; y \# \text{xvec}; y \# \text{yvec}; y \# M;$

distinct xvec; distinct yvec;

$\text{xvec} \#* \Psi; y \# \Psi; \text{yvec} \#* \Psi; \text{xvec} \#* P; y \# P;$

$\text{yvec} \#* P; \text{xvec} \#* M; y \# M;$

$\llbracket \text{yvec} \#* M; \text{xvec} \#* \text{yvec} \rrbracket \implies$

$\text{Prop } (M(\nu*(\text{xvec}@y\#\text{yvec})) \langle N \rangle) P'$

and *rBrOpen*: $\bigwedge M \text{ xvec yvec } y N P'. \llbracket \Psi \triangleright ((x, y) \cdot P) \mapsto iM(\nu*(\text{xvec}@yvec)) \langle N \rangle \prec P'; y \in \text{supp } N;$

$x \# N; x \# P'; x \neq y; y \# \text{xvec}; y \# \text{yvec}; y \# M;$

distinct xvec; distinct yvec;

$\text{xvec} \#* \Psi; y \# \Psi; \text{yvec} \#* \Psi; \text{xvec} \#* P; y \# P;$

$\text{yvec} \#* P; \text{xvec} \#* M; y \# M;$

$\llbracket \text{yvec} \#* M; \text{xvec} \#* \text{yvec} \rrbracket \implies$

$\text{Prop } (iM(\nu*(\text{xvec}@y\#\text{yvec})) \langle N \rangle) P'$

and *rScope*: $\bigwedge P'. \llbracket \Psi \triangleright P \mapsto \alpha \prec P' \rrbracket \implies \text{Prop } \alpha ((\nu x)P')$

and *rBrClose*: $\bigwedge M \text{ xvec } N P'.$

$\llbracket \Psi \triangleright P \mapsto iM(\nu*\text{xvec}) \langle N \rangle \prec P';$

$x \in \text{supp } M;$

distinct xvec; xvec $\#* \Psi; \text{xvec} \#* P;$

$\text{xvec} \#* M;$

$x \# \Psi; x \# \text{xvec};$

$\text{xvec} \#* C \rrbracket \implies \text{Prop } (\tau) ((\nu x)((\nu*\text{xvec})P'))$

shows *Prop* $\alpha P'$

<proof>

lemma *resInputCases'*[*consumes 4, case-names cRes*]:

fixes Ψ :: 'b

and x :: name

and M :: 'a

and N $:: 'a$
and P $:: ('a, 'b, 'c) psi$
and R $:: ('a, 'b, 'c) psi$
and C $:: 'f::fs-name$

assumes $Trans: \Psi \triangleright (\nu x)P \mapsto M(N) \prec R$
and $2: x \# \Psi$
and $x \# (M(N))$
and $4: x \# R$
and $rScope: \bigwedge P'. [\Psi \triangleright P \mapsto M(N) \prec P'] \implies Prop ((\nu x)P')$

shows $Prop R$
 $\langle proof \rangle$

lemma $resBrInputCases'[consumes 4, case-names cRes]:$

fixes Ψ $:: 'b$
and x $:: name$
and M $:: 'a$
and N $:: 'a$
and P $:: ('a, 'b, 'c) psi$
and R $:: ('a, 'b, 'c) psi$
and C $:: 'f::fs-name$

assumes $Trans: \Psi \triangleright (\nu x)P \mapsto_i M(N) \prec R$
and $2: x \# \Psi$
and $x \# ({}_i M(N))$
and $4: x \# R$
and $rScope: \bigwedge P'. [\Psi \triangleright P \mapsto_i M(N) \prec P'] \implies Prop ((\nu x)P')$

shows $Prop R$
 $\langle proof \rangle$

lemma $resOutputCases''[consumes 7, case-names cOpen cRes]:$

fixes Ψ $:: 'b$
and x $:: name$
and M $:: 'a$
and N $:: 'a$
and P $:: ('a, 'b, 'c) psi$
and R $:: ('a, 'b, 'c) psi$
and C $:: 'f::fs-name$

assumes $Trans: \Psi \triangleright (\nu x)P \mapsto M(\nu*(zvec1 @ zvec2))(N) \prec R$
and $1: x \# \Psi$
and $x \# (M(\nu*(zvec1 @ zvec2))(N))$
and $3: x \# R$
and $(zvec1 @ zvec2) \#* \Psi$
and $(zvec1 @ zvec2) \#* P$
and $(zvec1 @ zvec2) \#* M$
and $rOpen: \bigwedge M xvec yvec y N P'. [\Psi \triangleright ((x, y) \cdot P) \mapsto M(\nu*(xvec@yvec))(N)]$

$\prec P'; y \in \text{supp } N;$
 $x \# N; x \# P'; x \neq y; y \# \text{xvec}; y \# \text{yvec}; y \# M;$
 $\text{distinct } \text{xvec}; \text{distinct } \text{yvec};$
 $\text{yvec} \#* P; \text{xvec} \#* M; y \# M;$
 $\text{xvec} \#* \Psi; y \# \Psi; \text{yvec} \#* \Psi; \text{xvec} \#* P; y \# P;$
 $\text{yvec} \#* M; \text{xvec} \#* \text{yvec}] \implies$
 $\text{Prop } P'$
and $rScope: \bigwedge P'. [\Psi \triangleright P \mapsto M(\nu*(\text{zvec1} @ \text{zvec2})) \langle N \rangle \prec P] \implies \text{Prop}$
 $(\langle \nu x \rangle P')$

shows $\text{Prop } R$
 $\langle \text{proof} \rangle$

lemma resOutputCases'' [consumes 7, case-names $cOpen$ $cRes$]:

fixes $\Psi :: 'b$
and $x :: \text{name}$
and $z :: \text{name}$
and $M :: 'a$
and $N :: 'a$
and $P :: ('a, 'b, 'c) \text{psi}$
and $R :: ('a, 'b, 'c) \text{psi}$
and $C :: 'f::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright (\nu x)P \mapsto M(\nu*(\text{zvec1} @ y \# \text{zvec2})) \langle N \rangle \prec R$
and $1: x \# \Psi$
and $x \# (M(\nu*(\text{zvec1} @ y \# \text{zvec2})) \langle N \rangle)$
and $3: x \# R$
and $(\text{zvec1} @ y \# \text{zvec2}) \#* \Psi$
and $(\text{zvec1} @ y \# \text{zvec2}) \#* P$
and $(\text{zvec1} @ y \# \text{zvec2}) \#* M$
and $rOpen: \bigwedge M \text{xvec } \text{yvec } y N P'. [\Psi \triangleright ((x, y) \cdot P) \mapsto M(\nu*(\text{xvec} @ \text{yvec})) \langle N \rangle$
 $\prec P'; y \in \text{supp } N;$
 $x \# N; x \# P'; x \neq y; y \# \text{xvec}; y \# \text{yvec}; y \# M;$
 $\text{distinct } \text{xvec}; \text{distinct } \text{yvec};$
 $\text{yvec} \#* P; \text{xvec} \#* M; y \# M;$
 $\text{xvec} \#* \Psi; y \# \Psi; \text{yvec} \#* \Psi; \text{xvec} \#* P; y \# P;$
 $\text{yvec} \#* M; \text{xvec} \#* \text{yvec}] \implies$
 $\text{Prop } P'$
and $rScope: \bigwedge P'. [\Psi \triangleright P \mapsto M(\nu*(\text{zvec1} @ y \# \text{zvec2})) \langle N \rangle \prec P] \implies \text{Prop}$
 $(\langle \nu x \rangle P')$

shows $\text{Prop } R$
 $\langle \text{proof} \rangle$

abbreviation

$\text{statImpJudge } (- \leftrightarrow - [80, 80] 80)$
where $\Psi \leftrightarrow \Psi' \equiv \text{Assertion.StatImp } \Psi \Psi'$

lemma statEqTransition :


```

fixes  $\Psi :: 'b$ 
  and  $P :: ('a, 'b, 'c) psi$ 
  and  $Rs :: ('a, 'b, 'c) residual$ 
  and  $\Psi' :: 'b$ 

assumes  $\Psi \triangleright P \mapsto Rs$ 
  and  $\Psi \simeq \Psi'$ 

shows  $\Psi' \triangleright P \mapsto Rs$ 
   $\langle proof \rangle$ 

lemma brInputTermSupp:
  fixes  $\Psi :: 'b::fs-name$ 
    and  $P :: ('a, 'b, ('c::fs-name)) psi$ 
    and  $P' :: ('a, 'b, 'c) psi$ 
    and  $N :: 'a::fs-name$ 
    and  $K :: 'a::fs-name$ 

assumes  $\Psi \triangleright P \mapsto \downarrow K(\downarrow N) \prec P'$ 

shows  $(supp K) \subseteq ((supp P)::name set)$ 
   $\langle proof \rangle$ 

lemma brOutputTermSupp:
  fixes  $\Psi :: 'b::fs-name$ 
    and  $P :: ('a, 'b, ('c::fs-name)) psi$ 
    and  $P' :: ('a, 'b, 'c) psi$ 
    and  $N :: 'a::fs-name$ 
    and  $K :: 'a::fs-name$ 
    and  $xvec :: name list$ 

assumes  $\Psi \triangleright P \mapsto RBrOut K ((\nu*xvec)N \prec' P')$ 

shows  $(supp K) \subseteq ((supp P)::name set)$ 
   $\langle proof \rangle$ 

lemma actionPar1Dest':
  fixes  $\alpha :: ('a::fs-name) action$ 
    and  $P :: ('a, 'b::fs-name, 'c::fs-name) psi$ 
    and  $\beta :: ('a::fs-name) action$ 
    and  $Q :: ('a, 'b, 'c) psi$ 
    and  $R :: ('a, 'b, 'c) psi$ 

assumes  $\alpha \prec P = \beta \prec (Q \parallel R)$ 
  and  $bn \alpha \#* R$ 
  and  $bn \beta \#* R$ 

obtains  $T$  where  $P = T \parallel R$  and  $\alpha \prec T = \beta \prec Q$ 
   $\langle proof \rangle$ 

```

lemma *actionPar2Dest'*:
fixes $\alpha :: ('a::fs\text{-name})\ action$
and $P :: ('a, 'b::fs\text{-name}, 'c::fs\text{-name})\ psi$
and $\beta :: ('a::fs\text{-name})\ action$
and $Q :: ('a, 'b, 'c)\ psi$
and $R :: ('a, 'b, 'c)\ psi$

assumes $\alpha \prec P = \beta \prec (Q \parallel R)$
and $bn\ \alpha \#^* Q$
and $bn\ \beta \#^* Q$

obtains T **where** $P = Q \parallel T$ **and** $\alpha \prec T = \beta \prec R$
 $\langle proof \rangle$

lemma *expandNonTauFrame*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c)\ psi$
and $\alpha :: 'a\ action$
and $P' :: ('a, 'b, 'c)\ psi$
and $A_P :: name\ list$
and $\Psi_P :: 'b$
and $C :: 'f::fs\text{-name}$
and $C' :: 'g::fs\text{-name}$

assumes *Trans*: $\Psi \triangleright P \mapsto \alpha \prec P'$
and $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and $distinct\ A_P$
and $bn\ \alpha \#^* subject\ \alpha$
and $A_P \#^* P$
and $A_P \#^* \alpha$
and $A_P \#^* C$
and $A_P \#^* C'$
and $bn\ \alpha \#^* P$
and $bn\ \alpha \#^* C'$
and $\alpha \neq \tau$

obtains $p\ \Psi'\ A_P'\ \Psi_P'$ **where** $set\ p \subseteq set(bn\ \alpha) \times set(bn(p \cdot \alpha))$ **and** $(p \cdot \Psi_P)$
 $\otimes \Psi' \simeq \Psi_P'$ **and** $distinctPerm\ p$ **and**
 $extractFrame\ P' = \langle A_P', \Psi_P' \rangle$ **and** $A_P' \#^* P'$ **and** $A_P' \#^* \alpha$ **and** $A_P' \#^* (p \cdot \alpha)$
and
 $A_P' \#^* C$ **and** $(bn(p \cdot \alpha)) \#^* C'$ **and** $(bn(p \cdot \alpha)) \#^* \alpha$ **and** $(bn(p \cdot \alpha)) \#^* P'$ **and**
 $distinct\ A_P'$
 $\langle proof \rangle$

lemma *expandTauFrame*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c)\ psi$
and $P' :: ('a, 'b, 'c)\ psi$

and A_P $::$ *name list*
and Ψ_P $::$ *'b*
and C $::$ *'f::fs-name*

assumes $\Psi \triangleright P \mapsto \tau \prec P'$
and $\text{extractFrame } P = \langle A_P, \Psi_P \rangle$
and $\text{distinct } A_P$
and $A_P \#^* P$
and $A_P \#^* C$

obtains $\Psi' A_{P'} \Psi_{P'}$ **where** $\text{extractFrame } P' = \langle A_{P'}, \Psi_{P'} \rangle$ **and** $\Psi_P \otimes \Psi' \simeq \Psi_{P'}$
and $A_{P'} \#^* C$ **and** $A_{P'} \#^* P'$ **and** $\text{distinct } A_{P'}$
\langle proof \rangle

lemma *expandFrame*:

fixes Ψ $::$ *'b*
and P $::$ *('a, 'b, 'c) psi*
and α $::$ *'a action*
and P' $::$ *('a, 'b, 'c) psi*
and A_P $::$ *name list*
and Ψ_P $::$ *'b*
and C $::$ *'f::fs-name*
and C' $::$ *'g::fs-name*

assumes *Trans*: $\Psi \triangleright P \mapsto \alpha \prec P'$
and $\text{extractFrame } P = \langle A_P, \Psi_P \rangle$
and $\text{distinct } A_P$
and $\text{bn } \alpha \#^* \text{subject } \alpha$
and $\text{distinct}(\text{bn } \alpha)$
and $A_P \#^* \alpha$
and $A_P \#^* P$
and $A_P \#^* C$
and $A_P \#^* C'$
and $\text{bn } \alpha \#^* P$
and $\text{bn } \alpha \#^* C'$

obtains $p \Psi' A_{P'} \Psi_{P'}$ **where** $\text{set } p \subseteq \text{set}(\text{bn } \alpha) \times \text{set}(\text{bn}(p \cdot \alpha))$ **and** $(p \cdot \Psi_P) \otimes \Psi' \simeq \Psi_{P'}$ **and** $\text{distinctPerm } p$ **and**
 $\text{extractFrame } P' = \langle A_{P'}, \Psi_{P'} \rangle$ **and** $A_{P'} \#^* P'$ **and** $A_{P'} \#^* \alpha$ **and** $A_{P'} \#^* (p \cdot \alpha)$
and
 $A_{P'} \#^* C$ **and** $(\text{bn}(p \cdot \alpha)) \#^* C'$ **and** $(\text{bn}(p \cdot \alpha)) \#^* \alpha$ **and** $(\text{bn}(p \cdot \alpha)) \#^* P'$ **and**
 $\text{distinct } A_{P'}$
\langle proof \rangle

abbreviation

$\text{frameImpJudge } (- \hookrightarrow_F - [80, 80] 80)$
where $F \hookrightarrow_F G \equiv \text{FrameStatImp } F G$

lemma *FrameStatEqImpCompose*:

```

fixes  $F :: 'b \text{ frame}$ 
  and  $G :: 'b \text{ frame}$ 
  and  $H :: 'b \text{ frame}$ 
  and  $I :: 'b \text{ frame}$ 

assumes  $F \simeq_F G$ 
  and  $G \hookrightarrow_F H$ 
  and  $H \simeq_F I$ 

shows  $F \hookrightarrow_F I$ 
   $\langle \text{proof} \rangle$ 

lemma transferNonTauFrame:
  fixes  $\Psi_F :: 'b$ 
    and  $P :: ('a, 'b, 'c) \text{ psi}$ 
    and  $\alpha :: 'a \text{ action}$ 
    and  $P' :: ('a, 'b, 'c) \text{ psi}$ 
    and  $A_F :: \text{name list}$ 
    and  $A_G :: \text{name list}$ 
    and  $\Psi_G :: 'b$ 

assumes  $\Psi_F \triangleright P \mapsto \alpha \prec P'$ 
  and  $\text{extractFrame } P = \langle A_P, \Psi_P \rangle$ 
  and  $\text{distinct } A_P$ 
  and  $\text{distinct}(bn \ \alpha)$ 
  and  $\langle A_F, \Psi_F \otimes \Psi_P \rangle \hookrightarrow_F \langle A_G, \Psi_G \otimes \Psi_P \rangle$ 
  and  $A_F \#^* P$ 
  and  $A_G \#^* P$ 
  and  $A_F \#^* \text{subject } \alpha$ 
  and  $A_G \#^* \text{subject } \alpha$ 
  and  $A_P \#^* A_F$ 
  and  $A_P \#^* A_G$ 
  and  $A_P \#^* \Psi_F$ 
  and  $A_P \#^* \Psi_G$ 
  and  $\alpha \neq \tau$ 

shows  $\Psi_G \triangleright P \mapsto \alpha \prec P'$ 
   $\langle \text{proof} \rangle$ 

lemma transferTauFrame:
  fixes  $\Psi_F :: 'b$ 
    and  $P :: ('a, 'b, 'c) \text{ psi}$ 
    and  $P' :: ('a, 'b, 'c) \text{ psi}$ 
    and  $A_F :: \text{name list}$ 
    and  $A_G :: \text{name list}$ 
    and  $\Psi_G :: 'b$ 

assumes  $\Psi_F \triangleright P \mapsto \tau \prec P'$ 
  and  $\text{extractFrame } P = \langle A_P, \Psi_P \rangle$ 

```

and *distinct* A_P
and $\langle A_F, \Psi_F \otimes \Psi_P \rangle \hookrightarrow_F \langle A_G, \Psi_G \otimes \Psi_P \rangle$
and $A_F \#^* P$
and $A_G \#^* P$
and $A_P \#^* A_F$
and $A_P \#^* A_G$
and $A_P \#^* \Psi_F$
and $A_P \#^* \Psi_G$

shows $\Psi_G \triangleright P \mapsto_{\tau} \prec P'$
<proof>

lemma *transferFrame*:

fixes $\Psi_F :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $\alpha :: 'a \text{ action}$
and $P' :: ('a, 'b, 'c) \text{ psi}$
and $A_F :: \text{ name list}$
and $A_G :: \text{ name list}$
and $\Psi_G :: 'b$

assumes $\Psi_F \triangleright P \mapsto_{\alpha} \prec P'$

and *extractFrame* $P = \langle A_P, \Psi_P \rangle$
and *distinct* A_P
and $\langle A_F, \Psi_F \otimes \Psi_P \rangle \hookrightarrow_F \langle A_G, \Psi_G \otimes \Psi_P \rangle$
and $A_F \#^* P$
and $A_G \#^* P$
and $A_F \#^* \text{subject } \alpha$
and $A_G \#^* \text{subject } \alpha$
and $A_P \#^* A_F$
and $A_P \#^* A_G$
and $A_P \#^* \Psi_F$
and $A_P \#^* \Psi_G$

shows $\Psi_G \triangleright P \mapsto_{\alpha} \prec P'$
<proof>

lemma *parCasesInputFrame*[*consumes* γ , *case-names* $cPar1$ $cPar2$]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $xvec :: \text{ name list}$
and $N :: 'a$
and $T :: ('a, 'b, 'c) \text{ psi}$
and $C :: 'd::\text{fs-name}$

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto_M(|N|) \prec T$

and *extractFrame*($P \parallel Q$) = $\langle A_{PQ}, \Psi_{PQ} \rangle$

and *distinct* A_{PQ}
and $A_{PQ} \#* \Psi$
and $A_{PQ} \#* P$
and $A_{PQ} \#* Q$
and $A_{PQ} \#* M$
and $rPar1: \bigwedge P' A_P \Psi_P A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(N) \prec P'; extractFrame$
 $P = \langle A_P, \Psi_P \rangle; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_P; distinct A_Q; A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies Prop (P' \parallel Q)$
and $rPar2: \bigwedge Q' A_P \Psi_P A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto M(N) \prec Q'; extractFrame$
 $P = \langle A_P, \Psi_P \rangle; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_P; distinct A_Q; A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies Prop (P \parallel Q')$
shows *Prop T*
\langle proof \rangle

lemma *parCasesBrInputFrame*[*consumes 7, case-names cPar1 cPar2 cBrMerge*]:

fixes Ψ $:: 'b$
and P $:: ('a, 'b, 'c) psi$
and Q $:: ('a, 'b, 'c) psi$
and M $:: 'a$
and $xvec$ $:: name list$
and N $:: 'a$
and T $:: ('a, 'b, 'c) psi$
and C $:: 'd::fs-name$

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto_i M(N) \prec T$
and $extractFrame(P \parallel Q) = \langle A_{PQ}, \Psi_{PQ} \rangle$
and *distinct* A_{PQ}
and $A_{PQ} \#* \Psi$
and $A_{PQ} \#* P$
and $A_{PQ} \#* Q$
and $A_{PQ} \#* M$
and $rPar1: \bigwedge P' A_P \Psi_P A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(N) \prec P'; extractFrame$
 $P = \langle A_P, \Psi_P \rangle; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_P; distinct A_Q; A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies Prop (P' \parallel Q)$
and $rPar2: \bigwedge Q' A_P \Psi_P A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M(N) \prec Q'; extractFrame$
 $P = \langle A_P, \Psi_P \rangle; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_P; distinct A_Q; A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies Prop (P \parallel Q')$

and *rBrMerge*: $\bigwedge \Psi_Q P' A_P \Psi_P Q' A_Q$.
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \iota M(N) \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle;$
distinct A_P ;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \iota M(N) \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_Q ;
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P;$
 $A_P \#* Q; A_P \#* A_Q;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P;$
 $A_Q \#* Q;$
 $A_{PQ} = A_P @ A_Q; \Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies$
 $\text{Prop } (P' \parallel Q')$
shows *Prop T*
<proof>

lemma *parCasesOutputFrame*[*consumes 11, case-names cPar1 cPar2*]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) *psi*
and Q :: ('a, 'b, 'c) *psi*
and M :: 'a
and *xvec* :: *name list*
and N :: 'a
and T :: ('a, 'b, 'c) *psi*
and C :: 'd::*fs-name*

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto M(\nu * xvec) \langle N \rangle \prec T$

and $xvec \#* \Psi$
and $xvec \#* P$
and $xvec \#* Q$
and $xvec \#* M$
and $\text{extractFrame}(P \parallel Q) = \langle A_{PQ}, \Psi_{PQ} \rangle$
and *distinct* A_{PQ}
and $A_{PQ} \#* \Psi$
and $A_{PQ} \#* P$
and $A_{PQ} \#* Q$
and $A_{PQ} \#* M$
and *rPar1*: $\bigwedge P' A_P \Psi_P A_Q \Psi_Q$. $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec) \langle N \rangle \prec P';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_P ; *distinct* A_Q ; $A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies \text{Prop } (P' \parallel Q)$
and *rPar2*: $\bigwedge Q' A_P \Psi_P A_Q \Psi_Q$. $\llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto M(\nu * xvec) \langle N \rangle \prec Q';$
 $\text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle;$
distinct A_P ; *distinct* A_Q ; $A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies \text{Prop } (P \parallel Q')$
shows *Prop T*
<proof>

lemma *parCasesBrOutputFrame*[*consumes 11, case-names cPar1 cPar2 cBrComm1 cBrComm2*]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and Q :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and N :: 'a
and T :: ('a, 'b, 'c) psi
and C :: 'd::fs-name

assumes *Trans*: $\Psi \triangleright P \parallel Q \mapsto_i M(\nu*xvec)\langle N \rangle \prec T$

and $xvec \#* \Psi$
and $xvec \#* P$
and $xvec \#* Q$
and $xvec \#* M$
and $extractFrame(P \parallel Q) = \langle A_{PQ}, \Psi_{PQ} \rangle$
and $distinct A_{PQ}$
and $A_{PQ} \#* \Psi$
and $A_{PQ} \#* P$
and $A_{PQ} \#* Q$
and $A_{PQ} \#* M$
and *rPar1*: $\bigwedge P' A_P \Psi_P A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P';$
 $extractFrame P = \langle A_P, \Psi_P \rangle; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_P; distinct A_Q; A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies Prop (P' \parallel Q)$
and *rPar2*: $\bigwedge Q' A_P \Psi_P A_Q \Psi_Q. \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu*xvec)\langle N \rangle \prec Q';$
 $extractFrame P = \langle A_P, \Psi_P \rangle; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_P; distinct A_Q; A_P \#* \Psi; A_P \#* P; A_P \#*$
 $Q; A_P \#* M; A_Q \#* \Psi; A_Q \#* P; A_Q \#* Q; A_Q \#* M;$
 $A_P \#* \Psi_Q; A_Q \#* \Psi_P; A_P \#* A_Q; A_{PQ} = A_P @ A_Q;$
 $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies Prop (P \parallel Q')$
and *rBrComm1*: $\bigwedge \Psi_Q P' A_P \Psi_P Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle; distinct$
 $A_P;$
 $\Psi \otimes \Psi_P \triangleright Q \mapsto_i M(\nu*xvec)\langle N \rangle \prec Q'; extractFrame Q = \langle A_Q, \Psi_Q \rangle;$
 $distinct A_Q;$
 $distinct xvec;$
 $A_P \#* \Psi; A_P \#* \Psi_Q; A_P \#* P; A_P \#* Q; A_P \#* A_Q;$
 $A_Q \#* \Psi; A_Q \#* \Psi_P; A_Q \#* P; A_Q \#* Q;$
 $A_P \#* M; A_Q \#* M; xvec \#* M;$
 $xvec \#* \Psi; xvec \#* P; xvec \#* Q;$
 $A_{PQ} = A_P @ A_Q; \Psi_{PQ} = \Psi_P \otimes \Psi_Q \rrbracket \implies$
 $Prop (P' \parallel Q')$
and *rBrComm2*: $\bigwedge \Psi_Q P' A_P \Psi_P Q' A_Q.$
 $\llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto_i M(\nu*xvec)\langle N \rangle \prec P'; extractFrame P = \langle A_P, \Psi_P \rangle;$

distinct A_P ;
 $\Psi \otimes \Psi_P \triangleright Q \mapsto \text{iM}(N) \prec Q'$; *extractFrame* $Q = \langle A_Q, \Psi_Q \rangle$; *distinct*
 A_Q ;
distinct *xvec*;
 $A_P \#* \Psi$; $A_P \#* \Psi_Q$; $A_P \#* P$; $A_P \#* Q$; $A_P \#* A_Q$;
 $A_Q \#* \Psi$; $A_Q \#* \Psi_P$; $A_Q \#* P$; $A_Q \#* Q$;
 $A_P \#* M$; $A_Q \#* M$; *xvec* $\#* M$;
xvec $\#* \Psi$; *xvec* $\#* P$; *xvec* $\#* Q$;
 $A_{PQ} = A_P @ A_Q$; $\Psi_{PQ} = \Psi_P \otimes \Psi_Q \implies$
Prop $(P' \parallel Q')$
shows *Prop* T
<proof>

inductive *bangPred* :: ('a, 'b, 'c) *psi* \Rightarrow ('a, 'b, 'c) *psi* \Rightarrow *bool*
where
aux1: *bangPred* P (! P)
| *aux2*: *bangPred* P ($P \parallel$! P)

lemma *bangInduct*[*consumes 1*, *case-names cPar1 cPar2 cComm1 cComm2 cBrMerge*
cBrComm1 cBrComm2 cBang]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) *psi*
and R_s :: ('a, 'b, 'c) *residual*
and *Prop* :: 'd::*fs-name* \Rightarrow 'b \Rightarrow ('a, 'b, 'c) *psi* \Rightarrow ('a, 'b, 'c) *residual* \Rightarrow *bool*
and C :: 'd

assumes $\Psi \triangleright !P \mapsto R_s$

and *rPar1*: $\bigwedge \alpha P' C. \llbracket \Psi \triangleright P \mapsto \alpha \prec P'; \text{bn } \alpha \#* \Psi; \text{bn } \alpha \#* P; \text{bn } \alpha \#*$
subject $\alpha; \text{bn } \alpha \#* C; \text{distinct}(\text{bn } \alpha) \rrbracket \implies \text{Prop } C \Psi (P \parallel !P) (\alpha \prec (P' \parallel !P))$
and *rPar2*: $\bigwedge \alpha P' C. \llbracket \Psi \triangleright !P \mapsto \alpha \prec P'; \text{bn } \alpha \#* \Psi; \text{bn } \alpha \#* P; \text{bn } \alpha \#*$
subject $\alpha; \text{bn } \alpha \#* C; \text{distinct}(\text{bn } \alpha);$
 $\bigwedge C. \text{Prop } C \Psi (!P) (\alpha \prec P') \rrbracket \implies \text{Prop } C \Psi (P \parallel !P) (\alpha$
 $\prec (P \parallel P'))$
and *rComm1*: $\bigwedge M N P' K \text{xvec } P'' C. \llbracket \Psi \triangleright P \mapsto M(N) \prec P'; \Psi \triangleright !P$
 $\mapsto K(\nu * \text{xvec})(N) \prec P''; \bigwedge C. \text{Prop } C \Psi (!P) (K(\nu * \text{xvec})(N) \prec P''); \Psi \vdash M \leftrightarrow$
 $K;$
 $\text{xvec } \#* \Psi; \text{xvec } \#* P; \text{xvec } \#* M; \text{xvec } \#* K;$
 $\text{xvec } \#* C; \text{distinct } \text{xvec} \rrbracket \implies \text{Prop } C \Psi (P \parallel !P) (\tau \prec (\nu * \text{xvec})(P' \parallel P''))$
and *rComm2*: $\bigwedge M \text{xvec } N P' K P'' C. \llbracket \Psi \triangleright P \mapsto M(\nu * \text{xvec})(N) \prec P'; \Psi \triangleright$
 $!P \mapsto K(N) \prec P''; \bigwedge C. \text{Prop } C \Psi (!P) (K(N) \prec P''); \Psi \vdash M \leftrightarrow K;$
 $\text{xvec } \#* \Psi; \text{xvec } \#* P; \text{xvec } \#* M; \text{xvec } \#* K;$
 $\text{xvec } \#* C; \text{distinct } \text{xvec} \rrbracket \implies \text{Prop } C \Psi (P \parallel !P) (\tau \prec (\nu * \text{xvec})(P' \parallel P''))$
and *rBrMerge*: $\bigwedge M N P' P'' C. \llbracket \Psi \triangleright P \mapsto \text{iM}(N) \prec P'; \Psi \triangleright !P \mapsto \text{iM}(N)$
 $\prec P''; \bigwedge C. \text{Prop } C \Psi (!P) (\text{iM}(N) \prec P'') \rrbracket \implies$
 $\text{Prop } C \Psi (P \parallel !P) (\text{iM}(N) \prec (P' \parallel P''))$
and *rBrComm1*: $\bigwedge M N P' \text{xvec } P'' C. \llbracket \Psi \triangleright P \mapsto \text{iM}(N) \prec P'; \Psi \triangleright !P$
 $\mapsto \text{iM}(\nu * \text{xvec})(N) \prec P''; \bigwedge C. \text{Prop } C \Psi (!P) (\text{iM}(\nu * \text{xvec})(N) \prec P'');$
 $\text{xvec } \#* \Psi; \text{xvec } \#* P; \text{xvec } \#* M; \text{xvec } \#* C;$
 $\text{distinct } \text{xvec} \rrbracket \implies \text{Prop } C \Psi (P \parallel !P) (\text{iM}(\nu * \text{xvec})(N) \prec (P' \parallel P''))$

and $rBrComm2: \bigwedge M N P' xvec P'' C. [\Psi \triangleright P \mapsto_i M(\nu * xvec)\langle N \rangle \prec P'; \Psi \triangleright !P \mapsto_i M(\langle N \rangle) \prec P''; \bigwedge C. Prop C \Psi (!P) (\downarrow_i M(\langle N \rangle) \prec P''); xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* C; distinct xvec] \implies Prop C \Psi (P \parallel !P) (\downarrow_i M(\nu * xvec)\langle N \rangle \prec (P' \parallel P''))$
and $rBang: \bigwedge Rs C. [\Psi \triangleright P \parallel !P \mapsto Rs; \bigwedge C. Prop C \Psi (P \parallel !P) Rs; guarded P] \implies Prop C \Psi (!P) Rs$
shows $Prop C \Psi (!P) Rs$
 $\langle proof \rangle$

lemma $bangInputInduct[consumes 1, case-names cPar1 cPar2 cBang]:$

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $Prop :: 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow 'a \Rightarrow 'a \Rightarrow ('a, 'b, 'c) psi \Rightarrow bool$

assumes $\Psi \triangleright !P \mapsto M(\langle N \rangle) \prec P'$

and $rPar1: \bigwedge P'. \Psi \triangleright P \mapsto M(\langle N \rangle) \prec P' \implies Prop \Psi (P \parallel !P) M N (P' \parallel !P)$
and $rPar2: \bigwedge P'. [\Psi \triangleright !P \mapsto M(\langle N \rangle) \prec P'; Prop \Psi (!P) M N P'] \implies Prop \Psi (P \parallel !P) M N (P \parallel P')$

and $rBang: \bigwedge P'. [\Psi \triangleright P \parallel !P \mapsto M(\langle N \rangle) \prec P'; Prop \Psi (P \parallel !P) M N P'; guarded P] \implies Prop \Psi (!P) M N P'$

shows $Prop \Psi (!P) M N P'$

$\langle proof \rangle$

lemma $brBangInputInduct[consumes 1, case-names cPar1 cPar2 cBrMerge cBang]:$

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$
and $P' :: ('a, 'b, 'c) psi$
and $Prop :: 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow 'a \Rightarrow 'a \Rightarrow ('a, 'b, 'c) psi \Rightarrow bool$

assumes $\Psi \triangleright !P \mapsto_i M(\langle N \rangle) \prec P'$

and $rPar1: \bigwedge P'. \Psi \triangleright P \mapsto_i M(\langle N \rangle) \prec P' \implies Prop \Psi (P \parallel !P) M N (P' \parallel !P)$

and $rPar2: \bigwedge P'. [\Psi \triangleright !P \mapsto_i M(\langle N \rangle) \prec P'; Prop \Psi (!P) M N P'] \implies Prop \Psi (P \parallel !P) M N (P \parallel P')$

and $rBrMerge: \bigwedge P' P'' C. [\Psi \triangleright P \mapsto_i M(\langle N \rangle) \prec P'; \Psi \triangleright !P \mapsto_i M(\langle N \rangle) \prec P''; \bigwedge C. Prop \Psi (!P) M N P''] \implies Prop \Psi (P \parallel !P) M N (P' \parallel P'')$

and $rBang: \bigwedge P'. [\Psi \triangleright P \parallel !P \mapsto_i M(\langle N \rangle) \prec P'; Prop \Psi (P \parallel !P) M N P'; guarded P] \implies Prop \Psi (!P) M N P'$

shows $Prop \Psi (!P) M N P'$

$\langle proof \rangle$

lemma $bangOutputInduct[consumes 1, case-names cPar1 cPar2 cBang]:$

fixes $\Psi :: 'b$

and P :: ('a, 'b, 'c) psi
and M :: 'a
and $xvec$:: name list
and N :: 'a
and P' :: ('a, 'b, 'c) psi
and $Prop$:: 'd::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow 'a \Rightarrow ('a, 'b, 'c) boundOutput
 \Rightarrow bool
and C :: 'd

assumes $\Psi \triangleright !P \mapsto M(\nu*xvec)\langle N \rangle \prec P'$
and $rPar1$: $\bigwedge xvec N P' C. \llbracket \Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'; xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* C; distinct\ xvec \rrbracket \Longrightarrow Prop\ C\ \Psi\ (P \parallel !P)\ M\ ((\nu*xvec)\langle N \rangle \prec' (P' \parallel !P))$
and $rPar2$: $\bigwedge xvec N P' C. \llbracket \Psi \triangleright !P \mapsto M(\nu*xvec)\langle N \rangle \prec P'; \bigwedge C. Prop\ C\ \Psi\ (!P)\ M\ ((\nu*xvec)\langle N \rangle \prec' P'); xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* C; distinct\ xvec \rrbracket \Longrightarrow Prop\ C\ \Psi\ (P \parallel !P)\ M\ ((\nu*xvec)\langle N \rangle \prec' (P \parallel P'))$
and $rBang$: $\bigwedge B C. \llbracket \Psi \triangleright P \parallel !P \mapsto (ROut\ M\ B); \bigwedge C. Prop\ C\ \Psi\ (P \parallel !P)\ M\ B; guarded\ P \rrbracket \Longrightarrow Prop\ C\ \Psi\ (!P)\ M\ B$

shows $Prop\ C\ \Psi\ (!P)\ M\ ((\nu*xvec)\langle N \rangle \prec' P')$
<proof>

lemma *bangTauInduct*[consumes 1, case-names $cPar1\ cPar2\ cComm1\ cComm2\ cBang$]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and P' :: ('a, 'b, 'c) psi
and $Prop$:: 'd::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow ('a, 'b, 'c) psi \Rightarrow bool
and C :: 'd

assumes $\Psi \triangleright !P \mapsto_{\tau} \prec P'$
and $rPar1$: $\bigwedge P' C. \Psi \triangleright P \mapsto_{\tau} \prec P' \Longrightarrow Prop\ C\ \Psi\ (P \parallel !P)\ (P' \parallel !P)$
and $rPar2$: $\bigwedge P' C. \llbracket \Psi \triangleright !P \mapsto_{\tau} \prec P'; \bigwedge C. Prop\ C\ \Psi\ (!P)\ P' \rrbracket \Longrightarrow Prop\ C\ \Psi\ (P \parallel !P)\ (P \parallel P')$
and $rComm1$: $\bigwedge M N P' K xvec P'' C. \llbracket \Psi \triangleright P \mapsto M\langle N \rangle \prec P'; \Psi \triangleright !P \mapsto K(\nu*xvec)\langle N \rangle \prec P''; \Psi \vdash M \leftrightarrow K; xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \Longrightarrow Prop\ C\ \Psi\ (P \parallel !P)\ ((\nu*xvec)\langle P' \parallel P'' \rangle)$
and $rComm2$: $\bigwedge M N P' K xvec P'' C. \llbracket \Psi \triangleright P \mapsto M(\nu*xvec)\langle N \rangle \prec P'; \Psi \triangleright !P \mapsto K\langle N \rangle \prec P''; \Psi \vdash M \leftrightarrow K; xvec \#* \Psi; xvec \#* P; xvec \#* M; xvec \#* K; xvec \#* C \rrbracket \Longrightarrow Prop\ C\ \Psi\ (P \parallel !P)\ ((\nu*xvec)\langle P' \parallel P'' \rangle)$
and $rBang$: $\bigwedge P' C. \llbracket \Psi \triangleright P \parallel !P \mapsto_{\tau} \prec P'; \bigwedge C. Prop\ C\ \Psi\ (P \parallel !P)\ P'; guarded\ P \rrbracket \Longrightarrow Prop\ C\ \Psi\ (!P)\ P'$

shows $Prop\ C\ \Psi\ (!P)\ P'$
<proof>

lemma *bangInduct'*[*consumes 2, case-names cAlpha cPar1 cPar2 cComm1 cComm2 cBrMerge cBrComm1 cBrComm2 cBang*]:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and α :: 'a action
and P' :: ('a, 'b, 'c) psi
and $Prop$:: 'd::fs-name \Rightarrow 'b \Rightarrow ('a, 'b, 'c) psi \Rightarrow 'a action \Rightarrow ('a, 'b, 'c) psi
 \Rightarrow bool
and C :: 'd::fs-name

assumes $\Psi \triangleright !P \mapsto \alpha \prec P'$

and $bn \ \alpha \ \#* \ \text{subject } \alpha$
and $rAlpha$: $\bigwedge \alpha \ P' \ p \ C. \llbracket bn \ \alpha \ \#* \ \Psi; bn \ \alpha \ \#* \ P; bn \ \alpha \ \#* \ \text{subject } \alpha; bn \ \alpha \ \#* \ C;$
 $set \ p \subseteq set(bn \ \alpha) \times set(bn(p \cdot \alpha)); distinctPerm \ p;$
 $bn(p \cdot \alpha) \ \#* \ \alpha; bn(p \cdot \alpha) \ \#* \ P'; Prop \ C \ \Psi \ (P \parallel !P) \ \alpha$

$P \rrbracket \Rightarrow$

$Prop \ C \ \Psi \ (P \parallel !P) \ (p \cdot \alpha) \ (p \cdot P')$

and $rPar1$: $\bigwedge \alpha \ P' \ C.$
 $\llbracket \Psi \triangleright P \mapsto \alpha \prec P'; bn \ \alpha \ \#* \ \Psi; bn \ \alpha \ \#* \ P; bn \ \alpha \ \#* \ \text{subject } \alpha; bn \ \alpha$
 $\#* \ C; distinct(bn \ \alpha) \rrbracket \Rightarrow$
 $Prop \ C \ \Psi \ (P \parallel !P) \ \alpha \ (P' \parallel !P)$

and $rPar2$: $\bigwedge \alpha \ P' \ C.$
 $\llbracket \Psi \triangleright !P \mapsto \alpha \prec P'; \bigwedge C. Prop \ C \ \Psi \ (!P) \ \alpha \ P';$
 $bn \ \alpha \ \#* \ \Psi; bn \ \alpha \ \#* \ P; bn \ \alpha \ \#* \ \text{subject } \alpha; bn \ \alpha \ \#* \ C; distinct(bn$
 $\alpha) \rrbracket \Rightarrow$

$Prop \ C \ \Psi \ (P \parallel !P) \ \alpha \ (P \parallel P')$

and $rComm1$: $\bigwedge M \ N \ P' \ K \ xvec \ P'' \ C. \llbracket \Psi \triangleright P \mapsto M \langle N \rangle \prec P'; \Psi \triangleright !P$
 $\mapsto K \langle \nu * xvec \rangle \langle N \rangle \prec P''; \bigwedge C. Prop \ C \ \Psi \ (!P) \ (K \langle \nu * xvec \rangle \langle N \rangle) \ P''; \Psi \vdash M \leftrightarrow K;$
 $xvec \ \#* \ \Psi; xvec \ \#* \ P; xvec \ \#* \ M; xvec \ \#* \ K;$
 $xvec \ \#* \ C; distinct \ xvec \rrbracket \Rightarrow Prop \ C \ \Psi \ (P \parallel !P) \ (\tau) \ ((\nu * xvec) \langle P' \parallel P'' \rangle)$

and $rComm2$: $\bigwedge M \ xvec \ N \ P' \ K \ P'' \ C. \llbracket \Psi \triangleright P \mapsto M \langle \nu * xvec \rangle \langle N \rangle \prec P'; \Psi \triangleright$
 $!P \mapsto K \langle N \rangle \prec P''; \bigwedge C. Prop \ C \ \Psi \ (!P) \ (K \langle N \rangle) \ P''; \Psi \vdash M \leftrightarrow K;$
 $xvec \ \#* \ \Psi; xvec \ \#* \ P; xvec \ \#* \ M; xvec \ \#* \ K;$
 $xvec \ \#* \ C; distinct \ xvec \rrbracket \Rightarrow Prop \ C \ \Psi \ (P \parallel !P) \ (\tau) \ ((\nu * xvec) \langle P' \parallel P'' \rangle)$

and $rBrMerge$: $\bigwedge M \ N \ P' \ P'' \ C. \llbracket \Psi \triangleright P \mapsto iM \langle N \rangle \prec P'; \Psi \triangleright !P \mapsto iM \langle N \rangle$
 $\prec P''; \bigwedge C. Prop \ C \ \Psi \ (!P) \ (iM \langle N \rangle) \ P'' \rrbracket \Rightarrow$
 $Prop \ C \ \Psi \ (P \parallel !P) \ (iM \langle N \rangle) \ (P' \parallel P'')$

and $rBrComm1$: $\bigwedge M \ N \ P' \ xvec \ P'' \ C. \llbracket \Psi \triangleright P \mapsto iM \langle N \rangle \prec P'; \Psi \triangleright !P$
 $\mapsto iM \langle \nu * xvec \rangle \langle N \rangle \prec P''; \bigwedge C. Prop \ C \ \Psi \ (!P) \ (iM \langle \nu * xvec \rangle \langle N \rangle) \ P'';$
 $xvec \ \#* \ \Psi; xvec \ \#* \ P; xvec \ \#* \ M; xvec \ \#* \ C;$
 $distinct \ xvec \rrbracket \Rightarrow Prop \ C \ \Psi \ (P \parallel !P) \ (iM \langle \nu * xvec \rangle \langle N \rangle) \ (P' \parallel P'')$

and $rBrComm2$: $\bigwedge M \ N \ P' \ xvec \ P'' \ C. \llbracket \Psi \triangleright P \mapsto iM \langle \nu * xvec \rangle \langle N \rangle \prec P'; \Psi$
 $\triangleright !P \mapsto iM \langle N \rangle \prec P''; \bigwedge C. Prop \ C \ \Psi \ (!P) \ (iM \langle N \rangle) \ P'';$
 $xvec \ \#* \ \Psi; xvec \ \#* \ P; xvec \ \#* \ M; xvec \ \#* \ C;$
 $distinct \ xvec \rrbracket \Rightarrow Prop \ C \ \Psi \ (P \parallel !P) \ (iM \langle \nu * xvec \rangle \langle N \rangle) \ (P' \parallel P'')$

and $rBang$: $\bigwedge \alpha \ P' \ C.$
 $\llbracket \Psi \triangleright P \parallel !P \mapsto \alpha \prec P'; guarded \ P; \bigwedge C. Prop \ C \ \Psi \ (P \parallel !P) \ \alpha$
 $P'; guarded \ P; distinct(bn \ \alpha) \rrbracket \Rightarrow$
 $Prop \ C \ \Psi \ (!P) \ \alpha \ P'$

shows $Prop\ C\ \Psi\ (!P)\ \alpha\ P'$
 $\langle proof \rangle$

lemma *brCommInAuxTooMuch*:

fixes Ψ :: 'b
and Ψ_Q :: 'b
and R :: ('a, 'b, 'c) psi
and M :: 'a
and N :: 'a
and R' :: ('a, 'b, 'c) psi
and A_R :: name list
and Ψ_R :: 'b
and A_P :: name list
and Ψ_P :: 'b
and A_Q :: name list

assumes $RTrans$: $\Psi \otimes \Psi_Q \triangleright R \mapsto_{iM} (!N) \prec R'$

and FrR : $extractFrame\ R = \langle A_R, \Psi_R \rangle$
and $distinct\ A_R$
and $QimpP$: $\langle A_Q, (\Psi \otimes \Psi_Q) \otimes \Psi_R \rangle \hookrightarrow_F \langle A_P, (\Psi \otimes \Psi_P) \otimes \Psi_R \rangle$
and $A_R \#^* A_P$
and $A_R \#^* A_Q$
and $A_R \#^* \Psi$
and $A_R \#^* \Psi_P$
and $A_R \#^* \Psi_Q$
and $A_R \#^* (\Psi \otimes \Psi_Q)$
and $A_R \#^* R$
and $A_R \#^* M$
and $A_P \#^* R$
and $A_P \#^* M$
and $A_Q \#^* R$
and $A_Q \#^* M$

shows $\Psi \otimes \Psi_P \triangleright R \mapsto_{iM} (!N) \prec R'$
 $\langle proof \rangle$

lemma *brCommInAux*:

fixes Ψ :: 'b
and Ψ_Q :: 'b
and R :: ('a, 'b, 'c) psi
and M :: 'a
and N :: 'a
and R' :: ('a, 'b, 'c) psi
and A_R :: name list
and Ψ_R :: 'b
and A_P :: name list
and Ψ_P :: 'b
and A_Q :: name list

assumes $RTrans: \Psi \otimes \Psi_Q \triangleright R \mapsto_{iM} \langle N \rangle \prec R'$
and $FrR: extractFrame R = \langle A_R, \Psi_R \rangle$
and $distinct A_R$
and $QimpP: \langle A_Q, (\Psi \otimes \Psi_Q) \otimes \Psi_R \rangle \hookrightarrow_F \langle A_P, (\Psi \otimes \Psi_P) \otimes \Psi_R \rangle$
and $A_R \#^* A_P$
and $A_R \#^* A_Q$
and $A_R \#^* \Psi$
and $A_R \#^* \Psi_P$
and $A_R \#^* \Psi_Q$
and $A_R \#^* R$
and $A_R \#^* M$
and $A_P \#^* R$
and $A_P \#^* M$
and $A_Q \#^* R$
and $A_Q \#^* M$

shows $\Psi \otimes \Psi_P \triangleright R \mapsto_{iM} \langle N \rangle \prec R'$
 $\langle proof \rangle$

lemma $brCommOutAuxTooMuch:$

fixes $\Psi :: 'b$
and $\Psi_Q :: 'b$
and $R :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $xvec :: name list$
and $N :: 'a$
and $R' :: ('a, 'b, 'c) psi$
and $A_R :: name list$
and $\Psi_R :: 'b$
and $A_P :: name list$
and $\Psi_P :: 'b$
and $A_Q :: name list$

assumes $RTrans: \Psi \otimes \Psi_Q \triangleright R \mapsto RBrOut M ((\nu^* xvec) N \prec' R')$
and $FrR: extractFrame R = \langle A_R, \Psi_R \rangle$
and $distinct A_R$
and $QimpP: \langle A_Q, (\Psi \otimes \Psi_Q) \otimes \Psi_R \rangle \hookrightarrow_F \langle A_P, (\Psi \otimes \Psi_P) \otimes \Psi_R \rangle$
and $A_R \#^* A_P$
and $A_R \#^* A_Q$
and $A_R \#^* \Psi$
and $A_R \#^* \Psi_P$
and $A_R \#^* \Psi_Q$
and $A_R \#^* (\Psi \otimes \Psi_Q)$
and $A_R \#^* R$
and $A_R \#^* M$
and $A_P \#^* R$
and $A_P \#^* M$
and $A_Q \#^* R$
and $A_Q \#^* M$

shows $\Psi \otimes \Psi_P \triangleright R \mapsto \text{jM}(\nu * \text{xvec}) \langle N \rangle \prec R'$
<proof>

lemma *brCommOutAux*:

fixes Ψ :: 'b
and Ψ_Q :: 'b
and R :: ('a, 'b, 'c) *psi*
and M :: 'a
and xvec :: *name list*
and N :: 'a
and R' :: ('a, 'b, 'c) *psi*
and A_R :: *name list*
and Ψ_R :: 'b
and A_P :: *name list*
and Ψ_P :: 'b
and A_Q :: *name list*

assumes *RTrans*: $\Psi \otimes \Psi_Q \triangleright R \mapsto \text{jM}(\nu * \text{xvec}) \langle N \rangle \prec R'$

and *FrR*: *extractFrame* $R = \langle A_R, \Psi_R \rangle$
and *distinct* A_R
and *QimpP*: $\langle A_Q, (\Psi \otimes \Psi_Q) \otimes \Psi_R \rangle \hookrightarrow_F \langle A_P, (\Psi \otimes \Psi_P) \otimes \Psi_R \rangle$
and $A_R \#^* A_P$
and $A_R \#^* A_Q$
and $A_R \#^* \Psi$
and $A_R \#^* \Psi_P$
and $A_R \#^* \Psi_Q$
and $A_R \#^* R$
and $A_R \#^* M$
and $A_P \#^* R$
and $A_P \#^* M$
and $A_Q \#^* R$
and $A_Q \#^* M$

shows $\Psi \otimes \Psi_P \triangleright R \mapsto \text{jM}(\nu * \text{xvec}) \langle N \rangle \prec R'$
<proof>

lemma *comm1Aux*:

fixes Ψ :: 'b
and Ψ_Q :: 'b
and R :: ('a, 'b, 'c) *psi*
and K :: 'a
and xvec :: *name list*
and N :: 'a
and R' :: ('a, 'b, 'c) *psi*
and A_R :: *name list*
and Ψ_R :: 'b
and P :: ('a, 'b, 'c) *psi*
and M :: 'a

and L $:: 'a$
and P' $:: ('a, 'b, 'c)$ *psi*
and A_P $::$ *name list*
and Ψ_P $:: 'b$
and A_Q $::$ *name list*

assumes $RTrans$: $\Psi \otimes \Psi_Q \triangleright R \mapsto K(\nu*xvec)\langle N \rangle \prec R'$
and FrR : $extractFrame\ R = \langle A_R, \Psi_R \rangle$
and $PTrans$: $\Psi \otimes \Psi_R \triangleright P \mapsto M\langle L \rangle \prec P'$
and $MeqK$: $\Psi \otimes \Psi_Q \otimes \Psi_R \vdash M \leftrightarrow K$
and $PeqQ$: $\langle A_Q, (\Psi \otimes \Psi_Q) \otimes \Psi_R \rangle \hookrightarrow_F \langle A_P, (\Psi \otimes \Psi_P) \otimes \Psi_R \rangle$
and FrP : $extractFrame\ P = \langle A_P, \Psi_P \rangle$
and FrQ : $extractFrame\ Q = \langle A_Q, \Psi_Q \rangle$
and *distinct* A_P
and *distinct* A_R
and $A_R \#^* A_P$
and $A_R \#^* A_Q$
and $A_R \#^* \Psi$
and $A_R \#^* P$
and $A_R \#^* Q$
and $A_R \#^* R$
and $A_R \#^* K$
and $A_P \#^* \Psi$
and $A_P \#^* R$
and $A_P \#^* P$
and $A_P \#^* M$
and $A_Q \#^* R$
and $A_Q \#^* M$

obtains K' **where** $\Psi \otimes \Psi_P \triangleright R \mapsto K'(\nu*xvec)\langle N \rangle \prec R'$ **and** $\Psi \otimes \Psi_P \otimes \Psi_R \vdash M \leftrightarrow K'$ **and** $A_R \#^* K'$
<proof>

lemma *comm2Aux*:

fixes Ψ $:: 'b$
and Ψ_Q $:: 'b$
and R $:: ('a, 'b, 'c)$ *psi*
and K $:: 'a$
and N $:: 'a$
and R' $:: ('a, 'b, 'c)$ *psi*
and A_R $::$ *name list*
and Ψ_R $:: 'b$
and P $:: ('a, 'b, 'c)$ *psi*
and M $:: 'a$
and $xvec$ $::$ *name list*
and P' $:: ('a, 'b, 'c)$ *psi*
and A_P $::$ *name list*
and Ψ_P $:: 'b$
and A_Q $::$ *name list*


```

assumes RTrans:  $\Psi \otimes \Psi_Q \triangleright R \mapsto K(|N|) \prec R'$ 
and FrR: extractFrame  $R = \langle A_R, \Psi_R \rangle$ 
and PTrans:  $\Psi \otimes \Psi_R \triangleright P \mapsto M(|\nu * xvec|)(|N|) \prec P'$ 
and MeqK:  $\Psi \otimes \Psi_Q \otimes \Psi_R \vdash M \leftrightarrow K$ 
and QimpP:  $\langle A_Q, (\Psi \otimes \Psi_Q) \otimes \Psi_R \rangle \hookrightarrow_F \langle A_P, (\Psi \otimes \Psi_P) \otimes \Psi_R \rangle$ 
and FrP: extractFrame  $P = \langle A_P, \Psi_P \rangle$ 
and FrQ: extractFrame  $Q = \langle A_Q, \Psi_Q \rangle$ 
and distinct  $A_P$ 
and distinct  $A_R$ 
and  $A_R \#^* A_P$ 
and  $A_R \#^* A_Q$ 
and  $A_R \#^* \Psi$ 
and  $A_R \#^* P$ 
and  $A_R \#^* Q$ 
and  $A_R \#^* R$ 
and  $A_R \#^* K$ 
and  $A_P \#^* \Psi$ 
and  $A_P \#^* R$ 
and  $A_P \#^* P$ 
and  $A_P \#^* M$ 
and  $A_Q \#^* R$ 
and  $A_Q \#^* M$ 
and  $A_R \#^* xvec$ 
and  $xvec \#^* M$ 

```

```

obtains  $K'$  where  $\Psi \otimes \Psi_P \triangleright R \mapsto K'(|N|) \prec R'$  and  $\Psi \otimes \Psi_P \otimes \Psi_R \vdash M \leftrightarrow K'$ 
and  $A_R \#^* K'$ 
<proof>

```

end

end

theory *Simulation*

imports *Semantics*

begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Simulation* from [1].

context *env* **begin**

definition

```

simulation :: 'b  $\Rightarrow$  ('a, 'b, 'c) psi  $\Rightarrow$ 
  ('b  $\times$  ('a, 'b, 'c) psi  $\times$  ('a, 'b, 'c) psi) set  $\Rightarrow$ 
  ('a, 'b, 'c) psi  $\Rightarrow$  bool (-  $\triangleright$  -  $\rightsquigarrow$  [-] - [80, 80, 80, 80] 80)
```

where

```

 $\Psi \triangleright P \rightsquigarrow[Rel] Q \equiv \forall \alpha Q'. \Psi \triangleright Q \mapsto \alpha \prec Q' \longrightarrow bn \ \alpha \ \#^* \ \Psi \longrightarrow bn \ \alpha \ \#^* \ P$ 
 $\longrightarrow (\exists P'. \Psi \triangleright P \mapsto \alpha \prec P' \wedge (\Psi, P', Q') \in Rel)$ 
```

abbreviation

simulationNilJudge $(- \rightsquigarrow[-] - [80, 80, 80] 80)$ **where** $P \rightsquigarrow[Rel] Q \equiv SBottom'$
 $\triangleright P \rightsquigarrow[Rel] Q$

lemma *simI*[*consumes 1, case-names cSim*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Q :: ('a, 'b, 'c) psi$
and $C :: 'd::fs-name$

assumes *Eqvt*: *eqvt Rel*

and *Sim*: $\bigwedge \alpha Q'. [\Psi \triangleright Q \mapsto \alpha \prec Q'; bn \alpha \#* P; bn \alpha \#* Q; bn \alpha \#* \Psi;$
distinct($bn \alpha$);

$bn \alpha \#* (subject \alpha); bn \alpha \#* C] \implies \exists P'. \Psi \triangleright P \mapsto \alpha \prec P'$
 $\wedge (\Psi, P', Q') \in Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] Q$

<proof>

lemma *simI2*[*case-names cSim*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Q :: ('a, 'b, 'c) psi$
and $C :: 'd::fs-name$

assumes *Sim*: $\bigwedge \alpha Q'. [\Psi \triangleright Q \mapsto \alpha \prec Q'; bn \alpha \#* P; bn \alpha \#* \Psi; distinct(bn \alpha)]$
 $\implies \exists P'. \Psi \triangleright P \mapsto \alpha \prec P' \wedge (\Psi, P', Q') \in Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] Q$

<proof>

lemma *simIChainFresh*[*consumes 4, case-names cSim*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Q :: ('a, 'b, 'c) psi$
and *xvec* :: *name list*
and $C :: 'd::fs-name$

assumes *Eqvt*: *eqvt Rel*

and *xvec* $\#* \Psi$

and *xvec* $\#* P$

and *xvec* $\#* Q$

and *Sim*: $\bigwedge \alpha Q'. [\Psi \triangleright Q \mapsto \alpha \prec Q'; bn \alpha \#* P; bn \alpha \#* Q; bn \alpha \#* \Psi;$
 $bn \alpha \#* subject \alpha; distinct(bn \alpha); bn \alpha \#* C; xvec \#* \alpha; xvec$

$\#* Q'] \implies$

$\exists P'. \Psi \triangleright P \mapsto \alpha \prec P' \wedge (\Psi, P', Q') \in Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] Q$
 ⟨proof⟩

lemma *simIFresh*[consumes 4, case-names cSim]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Q :: ('a, 'b, 'c) psi$
and $x :: name$
and $C :: 'd::fs-name$

assumes *Eqvt*: *eqvt Rel*

and $x \# \Psi$
and $x \# P$
and $x \# Q$
and $\bigwedge \alpha Q'. [\Psi \triangleright Q \mapsto \alpha \prec Q'; bn \alpha \#* P; bn \alpha \#* Q; bn \alpha \#* \Psi;$
 $bn \alpha \#* subject \alpha; distinct(bn \alpha); bn \alpha \#* C; x \# \alpha; x \# Q'] \implies$
 $\exists P'. \Psi \triangleright P \mapsto \alpha \prec P' \wedge (\Psi, P', Q') \in Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] Q$
 ⟨proof⟩

lemma *simE*:

fixes $F :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Q :: ('a, 'b, 'c) psi$

assumes $\Psi \triangleright P \rightsquigarrow[Rel] Q$

shows $\bigwedge \alpha Q'. [\Psi \triangleright Q \mapsto \alpha \prec Q'; bn \alpha \#* \Psi; bn \alpha \#* P] \implies \exists P'. \Psi \triangleright P \mapsto \alpha$
 $\prec P' \wedge (\Psi, P', Q') \in Rel$
 ⟨proof⟩

lemma *simClosedAux*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Q :: ('a, 'b, 'c) psi$
and $p :: name prm$

assumes *EqvtRel*: *eqvt Rel*

and *PSimQ*: $\Psi \triangleright P \rightsquigarrow[Rel] Q$

shows $(p \cdot \Psi) \triangleright (p \cdot P) \rightsquigarrow[Rel] (p \cdot Q)$
 ⟨proof⟩

lemma *simClosed*:

fixes $\Psi :: 'b$

and $P :: ('a, 'b, 'c) \text{ psi}$
and $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $p :: \text{ name prm}$

assumes $EqvRel: \text{ eqv } Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] Q \implies (p \cdot \Psi) \triangleright (p \cdot P) \rightsquigarrow[Rel] (p \cdot Q)$
and $P \rightsquigarrow[Rel] Q \implies (p \cdot P) \rightsquigarrow[Rel] (p \cdot Q)$
 $\langle \text{proof} \rangle$

lemma *reflexive*:

fixes $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$
and $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $\{(\Psi, P, P) \mid \Psi P. \text{ True}\} \subseteq Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] P$
 $\langle \text{proof} \rangle$

lemma *transitive*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $Rel' :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$
and $R :: ('a, 'b, 'c) \text{ psi}$
and $Rel'' :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

assumes $PSimQ: \Psi \triangleright P \rightsquigarrow[Rel] Q$

and $QSimR: \Psi \triangleright Q \rightsquigarrow[Rel'] R$

and $Eqv: \text{ eqv } Rel''$

and $Set: \{(\Psi, P, R) \mid \Psi P R. \exists Q. (\Psi, P, Q) \in Rel \wedge (\Psi, Q, R) \in Rel'\} \subseteq Rel''$

shows $\Psi \triangleright P \rightsquigarrow[Rel'] R$
 $\langle \text{proof} \rangle$

lemma *statEqSim*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $\Psi' :: 'b$

assumes $PSimQ: \Psi \triangleright P \rightsquigarrow[Rel] Q$

and $\text{ eqv } Rel'$

and $\Psi \simeq \Psi'$

```

and C1:  $\bigwedge \Psi'' R S \Psi''' . \llbracket (\Psi'' , R, S) \in \text{Rel}; \Psi'' \simeq \Psi''' \rrbracket \implies (\Psi''' , R, S) \in \text{Rel}'$ 

shows  $\Psi' \triangleright P \rightsquigarrow[\text{Rel}'] Q$ 
   $\langle \text{proof} \rangle$ 

lemma monotonic:
  fixes  $\Psi :: 'b$ 
    and  $P :: ('a, 'b, 'c) \text{psi}$ 
    and  $A :: ('b \times ('a, 'b, 'c) \text{psi} \times ('a, 'b, 'c) \text{psi}) \text{set}$ 
    and  $Q :: ('a, 'b, 'c) \text{psi}$ 
    and  $B :: ('b \times ('a, 'b, 'c) \text{psi} \times ('a, 'b, 'c) \text{psi}) \text{set}$ 

assumes  $\Psi \triangleright P \rightsquigarrow[A] Q$ 
  and  $A \subseteq B$ 

shows  $\Psi \triangleright P \rightsquigarrow[B] Q$ 
   $\langle \text{proof} \rangle$ 

end

end

theory Bisimulation
  imports Simulation
begin

  This file is a (heavily modified) variant of the theory Psi_Calculi.Bisimulation from [1].

context env begin

lemma monoCoinduct:  $\bigwedge x y xa xb xc P Q \Psi .$ 
  
$$x \leq y \implies$$

  
$$(\Psi \triangleright Q \rightsquigarrow[\{(xc, xb, xa). x xc xb xa\}] P) \longrightarrow$$

  
$$(\Psi \triangleright Q \rightsquigarrow[\{(xb, xa, xc). y xb xa xc\}] P)$$

   $\langle \text{proof} \rangle$ 

coinductive-set bisim ::  $('b \times ('a, 'b, 'c) \text{psi} \times ('a, 'b, 'c) \text{psi}) \text{set}$ 
  where
  
$$\text{step: } \llbracket (\text{insertAssertion } (\text{extractFrame } P)) \Psi \simeq_F (\text{insertAssertion } (\text{extractFrame } Q) \Psi) \rrbracket;$$

  
$$\Psi \triangleright P \rightsquigarrow[\text{bisim}] Q;$$

  
$$\forall \Psi' . (\Psi \otimes \Psi', P, Q) \in \text{bisim}; (\Psi, Q, P) \in \text{bisim} \rrbracket \implies (\Psi, P, Q) \in \text{bisim}$$

monos monoCoinduct

abbreviation
   $\text{bisimJudge } (- \triangleright - \rightsquigarrow - [70, 70, 70] 65) \text{ where } \Psi \triangleright P \rightsquigarrow Q \equiv (\Psi, P, Q) \in \text{bisim}$ 
abbreviation
   $\text{bisimNilJudge } (- \rightsquigarrow - [70, 70] 65) \text{ where } P \rightsquigarrow Q \equiv \text{SBottom}' \triangleright P \rightsquigarrow Q$ 

lemma bisimCoinductAux[consumes 1]:

```

fixes $F :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $X :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

assumes $(\Psi, P, Q) \in X$
and $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies \text{insertAssertion } (\text{extractFrame } P) \Psi \simeq_F$
 $\text{insertAssertion } (\text{extractFrame } Q) \Psi \wedge$
 $(\Psi \triangleright P \rightsquigarrow [(X \cup \text{bisim})] Q) \wedge$
 $(\forall \Psi'. (\Psi \otimes \Psi', P, Q) \in X \vee (\Psi \otimes \Psi', P, Q) \in$
 $\text{bisim}) \wedge$
 $((\Psi, Q, P) \in X \vee (\Psi, Q, P) \in \text{bisim})$

shows $(\Psi, P, Q) \in \text{bisim}$
 $\langle \text{proof} \rangle$

lemma $\text{bisimCoinduct}[\text{consumes } 1, \text{case-names } cStatEq \ cSim \ cExt \ cSym]$:
fixes $F :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $X :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

assumes $(\Psi, P, Q) \in X$
and $\bigwedge \Psi' R S. (\Psi', R, S) \in X \implies \text{insertAssertion } (\text{extractFrame } R) \Psi' \simeq_F$
 $\text{insertAssertion } (\text{extractFrame } S) \Psi'$
and $\bigwedge \Psi' R S. (\Psi', R, S) \in X \implies \Psi' \triangleright R \rightsquigarrow [(X \cup \text{bisim})] S$
and $\bigwedge \Psi' R S \Psi''. (\Psi', R, S) \in X \implies (\Psi' \otimes \Psi'', R, S) \in X \vee (\Psi' \otimes \Psi'', R,$
 $S) \in \text{bisim}$
and $\bigwedge \Psi' R S. (\Psi', R, S) \in X \implies (\Psi', S, R) \in X \vee (\Psi', S, R) \in \text{bisim}$

shows $(\Psi, P, Q) \in \text{bisim}$
 $\langle \text{proof} \rangle$

lemma $\text{bisimWeakCoinductAux}[\text{consumes } 1]$:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $X :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

assumes $(\Psi, P, Q) \in X$
and $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies \text{insertAssertion } (\text{extractFrame } P) \Psi \simeq_F$
 $\text{insertAssertion } (\text{extractFrame } Q) \Psi \wedge$
 $\Psi \triangleright P \rightsquigarrow [X] Q \wedge$
 $(\forall \Psi'. (\Psi \otimes \Psi', P, Q) \in X) \wedge (\Psi, Q, P) \in X$

shows $(\Psi, P, Q) \in \text{bisim}$
 $\langle \text{proof} \rangle$

lemma $\text{bisimWeakCoinduct}[\text{consumes } 1, \text{case-names } cStatEq \ cSim \ cExt \ cSym]$:

```

fixes  $F :: 'b$ 
  and  $P :: ('a, 'b, 'c) psi$ 
  and  $Q :: ('a, 'b, 'c) psi$ 
  and  $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$ 

assumes  $(\Psi, P, Q) \in X$ 
  and  $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies insertAssertion (extractFrame P) \Psi \simeq_F$ 
   $insertAssertion (extractFrame Q) \Psi$ 
  and  $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow[X] Q$ 
  and  $\bigwedge \Psi P Q \Psi'. (\Psi, P, Q) \in X \implies (\Psi \otimes \Psi', P, Q) \in X$ 
  and  $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies (\Psi, Q, P) \in X$ 

shows  $(\Psi, P, Q) \in bisim$ 
   $\langle proof \rangle$ 

lemma bisimE:
  fixes  $P :: ('a, 'b, 'c) psi$ 
  and  $Q :: ('a, 'b, 'c) psi$ 
  and  $\Psi :: 'b$ 
  and  $\Psi' :: 'b$ 

assumes  $(\Psi, P, Q) \in bisim$ 

shows  $insertAssertion (extractFrame P) \Psi \simeq_F insertAssertion (extractFrame Q)$ 
 $\Psi$ 
  and  $\Psi \triangleright P \rightsquigarrow[bisim] Q$ 
  and  $(\Psi \otimes \Psi', P, Q) \in bisim$ 
  and  $(\Psi, Q, P) \in bisim$ 
   $\langle proof \rangle$ 

lemma bisimI:
  fixes  $P :: ('a, 'b, 'c) psi$ 
  and  $Q :: ('a, 'b, 'c) psi$ 
  and  $\Psi :: 'b$ 

assumes  $insertAssertion (extractFrame P) \Psi \simeq_F insertAssertion (extractFrame$ 
 $Q) \Psi$ 
  and  $\Psi \triangleright P \rightsquigarrow[bisim] Q$ 
  and  $\forall \Psi'. (\Psi \otimes \Psi', P, Q) \in bisim$ 
  and  $(\Psi, Q, P) \in bisim$ 

shows  $(\Psi, P, Q) \in bisim$ 
   $\langle proof \rangle$ 

lemma bisimReflexive:
  fixes  $\Psi :: 'b$ 
  and  $P :: ('a, 'b, 'c) psi$ 

```

shows $\Psi \triangleright P \sim P$
<proof>

lemma *bisimClosed*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $p :: name prm$

assumes *PBisimQ*: $\Psi \triangleright P \sim Q$

shows $(p \cdot \Psi) \triangleright (p \cdot P) \sim (p \cdot Q)$
<proof>

lemma *bisimEqvt[simp]*:
shows *eqvt bisim*
<proof>

lemma *statEqBisim*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $\Psi' :: 'b$

assumes $\Psi \triangleright P \sim Q$
and $\Psi \simeq \Psi'$

shows $\Psi' \triangleright P \sim Q$
<proof>

lemma *bisimTransitive*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

assumes *PQ*: $\Psi \triangleright P \sim Q$
and *QR*: $\Psi \triangleright Q \sim R$

shows $\Psi \triangleright P \sim R$
<proof>

lemma *weakTransitiveCoinduct[case-names cStatEq cSim cExt cSym, case-conclusion bisim step, consumes 2]*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $p: (\Psi, P, Q) \in X$
and $Eqt: eqvt\ X$
and $rStatEq: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies insertAssertion\ (extractFrame\ P)\ \Psi$
 $\simeq_F\ insertAssertion\ (extractFrame\ Q)\ \Psi$
and $rSim: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow [(\{\Psi, P, Q\} \mid \Psi\ P\ P'\ Q'\ Q. \Psi \triangleright P \sim P' \wedge$
 $(\Psi, P', Q') \in X \wedge \Psi \triangleright Q' \sim Q)]\ Q$
and $rExt: \bigwedge \Psi\ P\ Q\ \Psi'. (\Psi, P, Q) \in X \implies (\Psi \otimes \Psi', P, Q) \in X$
and $rSym: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies (\Psi, Q, P) \in X$

shows $\Psi \triangleright P \sim Q$
 $\langle proof \rangle$

lemma $weakTransitiveCoinduct''[case-names\ cStatEq\ cSim\ cExt\ cSym, case-conclusion\ bisim\ step, consumes\ 2]:$

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c)\ psi$
and $Q :: ('a, 'b, 'c)\ psi$
and $X :: ('b \times ('a, 'b, 'c)\ psi \times ('a, 'b, 'c)\ psi)\ set$

assumes $p: (\Psi, P, Q) \in X$
and $Eqt: eqvt\ X$
and $rStatEq: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies insertAssertion\ (extractFrame\ P)\ \Psi$
 $\simeq_F\ insertAssertion\ (extractFrame\ Q)\ \Psi$
and $rSim: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow [(\{\Psi, P, Q\} \mid \Psi\ P\ P'\ Q'\ Q. \Psi \triangleright P \sim P' \wedge$
 $(\Psi, P', Q') \in X \wedge \Psi \triangleright Q' \sim Q)]\ Q$
and $rExt: \bigwedge \Psi\ P\ Q\ \Psi'. (\Psi, P, Q) \in X \implies (\Psi \otimes \Psi', P, Q) \in X$
and $rSym: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies$
 $(\Psi, Q, P) \in \{(\Psi, P, Q) \mid \Psi\ P\ P'\ Q'\ Q. \Psi \triangleright P \sim P' \wedge (\Psi, P', Q') \in X \wedge \Psi \triangleright Q' \sim Q\}$

shows $\Psi \triangleright P \sim Q$
 $\langle proof \rangle$

lemma $weakTransitiveCoinduct''[case-names\ cStatEq\ cSim\ cExt\ cSym, case-conclusion\ bisim\ step, consumes\ 2]:$

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c)\ psi$
and $Q :: ('a, 'b, 'c)\ psi$
and $X :: ('b \times ('a, 'b, 'c)\ psi \times ('a, 'b, 'c)\ psi)\ set$

assumes $p: (\Psi, P, Q) \in X$
and $Eqt: eqvt\ X$
and $rStatEq: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies insertAssertion\ (extractFrame\ P)\ \Psi$
 $\simeq_F\ insertAssertion\ (extractFrame\ Q)\ \Psi$
and $rSim: \bigwedge \Psi\ P\ Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow [(\{\Psi, P, Q\} \mid \Psi\ P\ P'\ Q'\ Q. \Psi \triangleright P \sim P' \wedge$

$\Psi \triangleright P \sim P' \wedge$

$(\Psi, P', Q') \in X \wedge$
 $\Psi \triangleright Q' \sim Q\}}] Q$

and $rExt: \bigwedge \Psi P Q \Psi'. (\Psi, P, Q) \in \{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge$
 $(\Psi, P', Q') \in X \wedge \Psi \triangleright Q' \sim Q\} \implies$
 $(\Psi \otimes \Psi', P, Q) \in \{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge$
 $(\Psi, P', Q') \in X \wedge \Psi \triangleright Q' \sim Q\}$

and $rSym: \bigwedge \Psi P Q. (\Psi, P, Q) \in \{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge$
 $(\Psi, P', Q') \in X \wedge \Psi \triangleright Q' \sim Q\} \implies$
 $(\Psi, Q, P) \in \{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge (\Psi, P',$
 $Q') \in X \wedge \Psi \triangleright Q' \sim Q\}$

shows $\Psi \triangleright P \sim Q$
 $\langle proof \rangle$

lemma *transitiveCoinduct*[*case-names* $cStatEq$ $cSim$ $cExt$ $cSym$, *case-conclusion*
bisim step, consumes 2]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $p: (\Psi, P, Q) \in X$

and $Eqvt: eqvt X$
and $rStatEq: \bigwedge \Psi' R S. (\Psi', R, S) \in X \implies insertAssertion (extractFrame R) \Psi'$
 $\simeq_F insertAssertion (extractFrame S) \Psi'$
and $rSim: \bigwedge \Psi' R S. (\Psi', R, S) \in X \implies \Psi' \triangleright R \rightsquigarrow [(\{\Psi', R, S\} \mid \Psi' R R' S'$
 $S. \Psi' \triangleright R \sim R' \wedge$

$(\Psi', R', S') \in X \vee \Psi' \triangleright$
 $R' \sim S') \wedge$

$\Psi' \triangleright S' \sim S\}}] S$

and $rExt: \bigwedge \Psi' R S \Psi''. (\Psi', R, S) \in X \implies (\Psi' \otimes \Psi'', R, S) \in X \vee \Psi' \otimes \Psi''$
 $\triangleright R \sim S$
and $rSym: \bigwedge \Psi' R S. (\Psi', R, S) \in X \implies (\Psi', S, R) \in X \vee \Psi' \triangleright S \sim R$

shows $\Psi \triangleright P \sim Q$
 $\langle proof \rangle$

lemma *transitiveCoinduct*[*case-names* $cStatEq$ $cSim$ $cExt$ $cSym$, *case-conclusion*
bisim step, consumes 2]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $p: (\Psi, P, Q) \in X$

and $Eqvt: eqvt X$
and $rStatEq: \bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies insertAssertion (extractFrame P) \Psi$

$\simeq_F \text{insertAssertion } (\text{extractFrame } Q) \Psi$
and $rSim: \bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow [(\{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge$
 $\Psi \triangleright P \sim P' \wedge$
 $(\Psi, P', Q') \in (X \cup \text{bisim}) \wedge$
 $\Psi \triangleright Q' \sim Q\}] Q$
and $rExt: \bigwedge \Psi P Q \Psi'. (\Psi, P, Q) \in X \implies (\Psi \otimes \Psi', P, Q) \in X \vee \Psi \otimes \Psi' \triangleright P$
 $\sim Q$
and $rSym: \bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies$
 $(\Psi, Q, P) \in \{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge ((\Psi, P',$
 $Q') \in (X \cup \text{bisim})) \wedge \Psi \triangleright Q' \sim Q\}$
shows $\Psi \triangleright P \sim Q$
 $\langle \text{proof} \rangle$

lemma *bisimSymmetric*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$

assumes $\Psi \triangleright P \sim Q$

shows $\Psi \triangleright Q \sim P$
 $\langle \text{proof} \rangle$

lemma *eqtTrans[intro]*:

assumes *eqt* X

shows *eqt* $\{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge ((\Psi, P', Q') \in X \vee \Psi \triangleright$
 $P' \sim Q') \wedge \Psi \triangleright Q' \sim Q\}$
 $\langle \text{proof} \rangle$

lemma *eqtWeakTrans[intro]*:

assumes *eqt* X

shows *eqt* $\{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge (\Psi, P', Q') \in X \wedge \Psi \triangleright Q'$
 $\sim Q\}$
 $\langle \text{proof} \rangle$

inductive-set

$\text{rel-trancl} :: ('b \times ('a, 'b, 'c) \text{psi} \times ('a, 'b, 'c) \text{psi}) \text{set} \implies ('b \times ('a, 'b, 'c) \text{psi} \times$
 $('a, 'b, 'c) \text{psi}) \text{set} \ ((-^*) [1000] 999)$

for $r :: ('b \times ('a, 'b, 'c) \text{psi} \times ('a, 'b, 'c) \text{psi}) \text{set}$

where

$r\text{-into-rel-trancl} [\text{intro}, \text{Pure.intro}]: (\Psi, P, Q) : r \implies (\Psi, P, Q) : r^*$
 $\mid \text{rel-trancl-into-rel-trancl} [\text{Pure.intro}]: (\Psi, P, Q) : r^* \implies (\Psi, Q, R) : r \implies$
 $(\Psi, P, R) : r^*$

lemma *rel-trancl-transitive*:

assumes $(\Psi, P, Q) \in \text{Rel}^*$

and $(\Psi, Q, R) \in Rel^*$
shows $(\Psi, P, R) \in Rel^*$
 $\langle proof \rangle$

lemma *rel-trancl-eqvt*:

assumes *eqvt* X
shows *eqvt* (X^*)
 $\langle proof \rangle$

lemma *bisimStarWeakCoinduct*[*consumes 2, case-names cStatEq cSim cExt cSym*]:

fixes $F :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $(\Psi, P, Q) \in X$

and *eqvt* X
and *rStatEq*: $\bigwedge \Psi' R S. (\Psi', R, S) \in X \implies insertAssertion (extractFrame R) \Psi' \simeq_F insertAssertion (extractFrame S) \Psi'$
and *rSim*: $\bigwedge \Psi' R S. (\Psi', R, S) \in X \implies \Psi' \triangleright R \rightsquigarrow[X^*] S$
and *rExt*: $\bigwedge \Psi' R S \Psi''. (\Psi', R, S) \in X \implies (\Psi' \otimes \Psi'', R, S) \in X$
and *rSym*: $\bigwedge \Psi' R S. (\Psi', R, S) \in X \implies (\Psi', S, R) \in X$

shows $(\Psi, P, Q) \in bisim$

$\langle proof \rangle$

lemma *weakTransitiveStarCoinduct*[*case-names cStatEq cSim cExt cSym, case-conclusion bisim step, consumes 2*]:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $p: (\Psi, P, Q) \in X$

and *Eqvt*: *eqvt* X
and *rStatEq*: $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies insertAssertion (extractFrame P) \Psi \simeq_F insertAssertion (extractFrame Q) \Psi$
and *rSim*: $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow[(\{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge$

$(\Psi, P', Q') \in X \wedge$
 $\Psi \triangleright Q' \sim Q\})^*] Q$

and *rExt*: $\bigwedge \Psi P Q \Psi'. (\Psi, P, Q) \in X \implies (\Psi \otimes \Psi', P, Q) \in X$
and *rSym*: $\bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies (\Psi, Q, P) \in X$

shows $\Psi \triangleright P \sim Q$

$\langle proof \rangle$

lemma *weakTransitiveStarCoinduct'*[*case-names cStatEq cSim cExt cSym, case-conclusion bisim step, consumes 2*]:

```

fixes  $\Psi :: 'b$ 
  and  $P :: ('a, 'b, 'c) psi$ 
  and  $Q :: ('a, 'b, 'c) psi$ 
  and  $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$ 

assumes  $p: (\Psi, P, Q) \in X$ 
  and  $Eqvt: eqvt X$ 
  and  $rStatEq: \bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies insertAssertion (extractFrame P) \Psi$ 
 $\simeq_F insertAssertion (extractFrame Q) \Psi$ 
  and  $rSim: \bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies \Psi \triangleright P \rightsquigarrow [(\{\Psi, P, Q\} \mid \Psi P P' Q' Q.$ 
 $\Psi \triangleright P \sim P' \wedge$ 

$$(\Psi, P', Q') \in X \wedge$$


$$\Psi \triangleright Q' \sim Q)]^* Q$$

  and  $rExt: \bigwedge \Psi P Q \Psi'. (\Psi, P, Q) \in X \implies (\Psi \otimes \Psi', P, Q) \in X$ 
  and  $rSym: \bigwedge \Psi P Q. (\Psi, P, Q) \in X \implies$ 

$$(\Psi, Q, P) \in \{(\Psi, P, Q) \mid \Psi P P' Q' Q. \Psi \triangleright P \sim P' \wedge (\Psi, P',$$

 $Q') \in X \wedge \Psi \triangleright Q' \sim Q\}$ 

shows  $\Psi \triangleright P \sim Q$ 
 $\langle proof \rangle$ 

lemma transitiveStarCoinduct[case-names cStatEq cSim cExt cSym, case-conclusion
bisim step, consumes 2]:
  fixes  $\Psi :: 'b$ 
  and  $P :: ('a, 'b, 'c) psi$ 
  and  $Q :: ('a, 'b, 'c) psi$ 
  and  $X :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$ 

assumes  $p: (\Psi, P, Q) \in X$ 
  and  $Eqvt: eqvt X$ 
  and  $rStatEq: \bigwedge \Psi' R S. (\Psi', R, S) \in X \implies insertAssertion (extractFrame R) \Psi'$ 
 $\simeq_F insertAssertion (extractFrame S) \Psi'$ 
  and  $rSim: \bigwedge \Psi' R S. (\Psi', R, S) \in X \implies \Psi' \triangleright R \rightsquigarrow [(\{\Psi', R, S\} \mid \Psi' R R' S'$ 
 $S. \Psi' \triangleright R \sim R' \wedge$ 

$$((\Psi', R', S') \in X \vee \Psi' \triangleright$$

 $R' \sim S') \wedge$ 

$$\Psi' \triangleright S' \sim S)]^* S$$

  and  $rExt: \bigwedge \Psi' R S \Psi''. (\Psi', R, S) \in X \implies (\Psi' \otimes \Psi'', R, S) \in X \vee \Psi' \otimes \Psi''$ 
 $\triangleright R \sim S$ 
  and  $rSym: \bigwedge \Psi' R S. (\Psi', R, S) \in X \implies (\Psi', S, R) \in X \vee \Psi' \triangleright S \sim R$ 

shows  $\Psi \triangleright P \sim Q$ 
 $\langle proof \rangle$ 

end

end

theory Sim-Pres

```

imports *Simulation*
begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Sim_Pres* from [1].

context *env* **begin**

lemma *inputPres*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) *psi*
and Rel :: ('b \times ('a, 'b, 'c) *psi* \times ('a, 'b, 'c) *psi*) *set*
and Q :: ('a, 'b, 'c) *psi*
and M :: 'a
and $xvec$:: *name list*
and N :: 'a

assumes $PRelQ$: $\bigwedge Tvec. length\ xvec = length\ Tvec \implies (\Psi, P[xvec::=Tvec], Q[xvec::=Tvec]) \in Rel$

shows $\Psi \triangleright M(\lambda*xvec\ N).P \rightsquigarrow[Rel] M(\lambda*xvec\ N).Q$
<proof>

lemma *outputPres*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) *psi*
and Rel :: ('b \times ('a, 'b, 'c) *psi* \times ('a, 'b, 'c) *psi*) *set*
and Q :: ('a, 'b, 'c) *psi*
and M :: 'a
and N :: 'a

assumes $PRelQ$: $(\Psi, P, Q) \in Rel$

shows $\Psi \triangleright M\langle N \rangle.P \rightsquigarrow[Rel] M\langle N \rangle.Q$
<proof>

lemma *casePres*:

fixes Ψ :: 'b
and CsP :: ('c \times ('a, 'b, 'c) *psi*) *list*
and Rel :: ('b \times ('a, 'b, 'c) *psi* \times ('a, 'b, 'c) *psi*) *set*
and CsQ :: ('c \times ('a, 'b, 'c) *psi*) *list*
and M :: 'a
and N :: 'a

assumes $PRelQ$: $\bigwedge \varphi\ Q. (\varphi, Q) \in set\ CsQ \implies \exists P. (\varphi, P) \in set\ CsP \wedge guarded\ P \wedge (\Psi, P, Q) \in Rel$

and Sim : $\bigwedge \Psi'\ R\ S. (\Psi', R, S) \in Rel \implies \Psi' \triangleright R \rightsquigarrow[Rel] S$
and $Rel \subseteq Rel'$

shows $\Psi \triangleright Cases\ CsP \rightsquigarrow[Rel'] Cases\ CsQ$

<proof>

lemma *resPres*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set
and Q :: ('a, 'b, 'c) psi
and x :: name
and Rel' :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set

assumes $PSimQ$: $\Psi \triangleright P \rightsquigarrow[Rel] Q$

and *eqvt* Rel'

and $x \# \Psi$

and $Rel \subseteq Rel'$

and $C1$: $\bigwedge \Psi' R S \text{ yvec. } \llbracket (\Psi', R, S) \in Rel; \text{yvec} \#* \Psi \rrbracket \implies (\Psi', (\nu * \text{yvec})R, (\nu * \text{yvec})S) \in Rel'$

shows $\Psi \triangleright (\nu x)P \rightsquigarrow[Rel'] (\nu x)Q$

<proof>

lemma *resChainPres*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set
and Q :: ('a, 'b, 'c) psi
and $xvec$:: name list

assumes $PSimQ$: $\Psi \triangleright P \rightsquigarrow[Rel] Q$

and *eqvt* Rel

and $xvec \#* \Psi$

and $C1$: $\bigwedge \Psi' R S \text{ yvec. } \llbracket (\Psi', R, S) \in Rel; \text{yvec} \#* \Psi \rrbracket \implies (\Psi', (\nu * \text{yvec})R, (\nu * \text{yvec})S) \in Rel$

shows $\Psi \triangleright (\nu * xvec)P \rightsquigarrow[Rel] (\nu * xvec)Q$

<proof>

lemma *parPres*:

fixes Ψ :: 'b
and P :: ('a, 'b, 'c) psi
and Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set
and Q :: ('a, 'b, 'c) psi
and R :: ('a, 'b, 'c) psi
and Rel' :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set

assumes $PRelQ$: $\bigwedge A_R \Psi_R. \llbracket \text{extractFrame } R = \langle A_R, \Psi_R \rangle; A_R \#* \Psi; A_R \#* P; A_R \#* Q \rrbracket \implies (\Psi \otimes \Psi_R, P, Q) \in Rel$

and *Eqvt*: *eqvt* Rel

and $Eqvt'$: *eqvt* Rel'

and *StatImp*: $\bigwedge \Psi' S T. (\Psi', S, T) \in Rel \implies insertAssertion (extractFrame T) \Psi' \hookrightarrow_F insertAssertion (extractFrame S) \Psi'$
and *Sim*: $\bigwedge \Psi' S T. (\Psi', S, T) \in Rel \implies \Psi' \triangleright S \rightsquigarrow[Rel] T$
and *Ext*: $\bigwedge \Psi' S T \Psi''. \llbracket (\Psi', S, T) \in Rel \rrbracket \implies (\Psi' \otimes \Psi'', S, T) \in Rel$

and *C1*: $\bigwedge \Psi' S T A_U \Psi_U U. \llbracket (\Psi' \otimes \Psi_U, S, T) \in Rel; extractFrame U = \langle A_U, \Psi_U \rangle; A_U \sharp^* \Psi'; A_U \sharp^* S; A_U \sharp^* T \rrbracket \implies (\Psi', S \parallel U, T \parallel U) \in Rel'$
and *C2*: $\bigwedge \Psi' S T xvec. \llbracket (\Psi', S, T) \in Rel'; xvec \sharp^* \Psi' \rrbracket \implies (\Psi', (\nu^* xvec)S, (\nu^* xvec)T) \in Rel'$
and *C3*: $\bigwedge \Psi' S T \Psi''. \llbracket (\Psi', S, T) \in Rel; \Psi' \simeq \Psi'' \rrbracket \implies (\Psi'', S, T) \in Rel$

shows $\Psi \triangleright P \parallel R \rightsquigarrow[Rel'] Q \parallel R$
<proof>

no-notation *relcomp* (**infixr** *O 75*)

lemma *bangPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $Rel' :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $(\Psi, P, Q) \in Rel$

and *eqt* Rel'
and *guarded* P
and *guarded* Q
and *cSim*: $\bigwedge \Psi' S T. (\Psi', S, T) \in Rel \implies \Psi' \triangleright S \rightsquigarrow[Rel] T$
and *cExt*: $\bigwedge \Psi' S T \Psi''. (\Psi', S, T) \in Rel \implies (\Psi' \otimes \Psi'', S, T) \in Rel$
and *cSym*: $\bigwedge \Psi' S T. (\Psi', S, T) \in Rel \implies (\Psi', T, S) \in Rel$
and *StatEq*: $\bigwedge \Psi' S T \Psi''. \llbracket (\Psi', S, T) \in Rel; \Psi' \simeq \Psi'' \rrbracket \implies (\Psi'', S, T) \in Rel$
and *Closed*: $\bigwedge \Psi' S T p. (\Psi', S, T) \in Rel \implies ((p::name prm) \cdot \Psi', p \cdot S, p \cdot T) \in Rel$
and *Assoc*: $\bigwedge \Psi' S T U. (\Psi', S \parallel (T \parallel U), (S \parallel T) \parallel U) \in Rel$
and *ParPres*: $\bigwedge \Psi' S T U. (\Psi', S, T) \in Rel \implies (\Psi', S \parallel U, T \parallel U) \in Rel$
and *FrameParPres*: $\bigwedge \Psi' \Psi_U S T U A_U. \llbracket (\Psi' \otimes \Psi_U, S, T) \in Rel; extractFrame U = \langle A_U, \Psi_U \rangle; A_U \sharp^* \Psi'; A_U \sharp^* S; A_U \sharp^* T \rrbracket \implies (\Psi', U \parallel S, U \parallel T) \in Rel$
and *ResPres*: $\bigwedge \Psi' S T xvec. \llbracket (\Psi', S, T) \in Rel; xvec \sharp^* \Psi' \rrbracket \implies (\Psi', (\nu^* xvec)S, (\nu^* xvec)T) \in Rel$
and *ScopeExt*: $\bigwedge xvec \Psi' S T. \llbracket xvec \sharp^* \Psi'; xvec \sharp^* T \rrbracket \implies (\Psi', (\nu^* xvec)(S \parallel T), ((\nu^* xvec)S) \parallel T) \in Rel$
and *Trans*: $\bigwedge \Psi' S T U. \llbracket (\Psi', S, T) \in Rel; (\Psi', T, U) \in Rel \rrbracket \implies (\Psi', S, U) \in Rel$
and *Compose*: $\bigwedge \Psi' S T U O. \llbracket (\Psi', S, T) \in Rel; (\Psi', T, U) \in Rel'; (\Psi', U, O) \in Rel \rrbracket \implies (\Psi', S, O) \in Rel'$
and *C1*: $\bigwedge \Psi S T U. \llbracket (\Psi, S, T) \in Rel; guarded S; guarded T \rrbracket \implies (\Psi, U \parallel !S,$

$U \parallel !T) \in Rel'$
and $Der: \bigwedge \Psi' S \alpha S' T. [\Psi' \triangleright !S \mapsto \alpha \prec S'; (\Psi', S, T) \in Rel; bn \alpha \#* \Psi';$
 $bn \alpha \#* S; bn \alpha \#* T; guarded T; bn \alpha \#* subject \alpha] \implies$
 $\exists T' U O. \Psi' \triangleright !T \mapsto \alpha \prec T' \wedge (\Psi', S', U \parallel !S)$
 $\in Rel \wedge (\Psi', T', O \parallel !T) \in Rel \wedge$
 $(\Psi', U, O) \in Rel \wedge ((supp U)::name set) \subseteq$
 $supp S' \wedge$
 $((supp O)::name set) \subseteq supp T'$

shows $\Psi \triangleright R \parallel !P \rightsquigarrow[Rel'] R \parallel !Q$
 $\langle proof \rangle$

notation *relcomp* (**infixr** *O* 75)

end

end

theory *Sim-Struct-Cong*

imports *Simulation HOL-Library.Multiset*

begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Sim_Struct_Cong* from [1].

lemma *partitionListLeft*:

assumes $xs@ys=xs'@y\#ys'$

and $y \in set xs$

and $distinct(xs@ys)$

obtains zs **where** $xs = xs'@y\#zs$ **and** $ys'=zs@ys$

$\langle proof \rangle$

lemma *partitionListRight*:

assumes $xs@ys=xs'@y\#ys'$

and $y \in set ys$

and $distinct(xs@ys)$

obtains zs **where** $xs' = xs@zs$ **and** $ys=zs@y\#ys'$

$\langle proof \rangle$

context *env* **begin**

lemma *resOutputCases''''*[*consumes* 8, *case-names* *cOpen cRes*]:

fixes Ψ $:: 'b$

and x $:: name$

and $zvec$ $:: name list$

and P $:: ('a, 'b, 'c) psi$

and α $:: 'a action$

and P' $:: ('a, 'b, 'c) psi$

and $C \quad :: 'f::fs\text{-name}$

assumes $Trans: \Psi \triangleright (\nu x)P \mapsto \alpha \prec P'$

and $1: x \# \Psi$
and $2: x \# \alpha$
and $3: x \# P'$
and $4: bn \alpha \#* \Psi$
and $5: bn \alpha \#* P$
and $6: bn \alpha \#* \text{subject } \alpha$
and $\alpha = M(\nu * zvec)\langle N \rangle$
and $rOpen: \bigwedge M \ xvec \ yvec \ y \ N \ P'. \llbracket \Psi \triangleright ((x, y) \cdot P) \mapsto M(\nu *(xvec @ yvec))\langle N \rangle \prec P'; y \in \text{supp } N; \quad x \# N; x \# P'; x \neq y; y \# xvec; y \# yvec; y \# M;$

distinct $xvec$; *distinct* $yvec$;
 $xvec \#* \Psi; y \# \Psi; yvec \#* \Psi; xvec \#* P; y \# P;$
 $yvec \#* P; xvec \#* M; y \# M;$

$yvec \#* M; xvec \#* yvec \rrbracket \implies$
 $Prop (M(\nu *(xvec @ yvec))\langle N \rangle) P'$

and $rScope: \bigwedge P'. \llbracket \Psi \triangleright P \mapsto \alpha \prec P' \rrbracket \implies Prop \alpha ((\nu x)P')$

shows $Prop \alpha P'$
 $\langle proof \rangle$

lemma $resOutputCases''''[consumes \gamma, case\text{-names } cOpen \ cRes]:$

fixes $\Psi \quad :: 'b$
and $x \quad :: name$
and $zvec \quad :: name \ list$
and $P \quad :: ('a, 'b, 'c) \ psi$
and $P' \quad :: ('a, 'b, 'c) \ psi$
and $C \quad :: 'f::fs\text{-name}$

assumes $Trans: \Psi \triangleright (\nu x)P \mapsto M(\nu * zvec)\langle N \rangle \prec P'$

and $1: x \# \Psi$
and $x \# M(\nu * zvec)\langle N \rangle$
and $3: x \# P'$
and $zvec \#* \Psi$
and $zvec \#* P$
and $zvec \#* M$
and $rOpen: \bigwedge M' \ xvec \ yvec \ y \ N' \ P'. \llbracket \Psi \triangleright ((x, y) \cdot P) \mapsto M'(\nu *(xvec @ yvec))\langle N' \rangle \prec P'; y \in \text{supp } N'; \quad x \# N'; x \# P'; x \neq y; y \# xvec; y \# yvec; y \# M';$

distinct $xvec$; *distinct* $yvec$;
 $xvec \#* \Psi; y \# \Psi; yvec \#* \Psi; xvec \#* P; y \# P;$
 $yvec \#* P; xvec \#* M'; y \# M';$

$yvec \#* M'; xvec \#* yvec; M'(\nu *(xvec @ yvec))\langle N' \rangle$
 $= M(\nu * zvec)\langle N \rangle \rrbracket \implies$
 $Prop P'$

and $rScope: \bigwedge P'. \llbracket \Psi \triangleright P \mapsto M(\nu * zvec)\langle N \rangle \prec P' \rrbracket \implies Prop ((\nu x)P')$

shows *Prop P'*
 ⟨*proof*⟩

lemma *resBrOutputCases'*[*consumes* γ , *case-names* *cBrOpen cRes*]:

fixes Ψ :: 'b
and *x* :: *name*
and *zvec* :: *name list*
and *P* :: ('a, 'b, 'c) *psi*
and *P'* :: ('a, 'b, 'c) *psi*
and *C* :: 'f::fs-name

assumes *Trans*: $\Psi \triangleright (\nu x)P \mapsto_{iM} (\nu * zvec) \langle N \rangle \prec P'$

and $1: x \# \Psi$
and $x \# iM(\nu * zvec) \langle N \rangle$
and $3: x \# P'$
and $zvec \#* \Psi$
and $zvec \#* P$
and $zvec \#* M$
and *rBrOpen*: $\bigwedge M' xvec yvec y N' P'. [\Psi \triangleright ((x, y) \cdot P) \mapsto_{iM'} (\nu * (xvec @ yvec)) \langle N' \rangle \prec P'; y \in \text{supp } N'];$

$x \# N'; x \# P'; x \neq y; y \# xvec; y \# yvec; y \# M';$

distinct xvec; distinct yvec;

$xvec \#* \Psi; y \# \Psi; yvec \#* \Psi; xvec \#* P; y \# P;$

$yvec \#* P; xvec \#* M'; y \# M';$

$yvec \#* M'; xvec \#* yvec; iM'(\nu * (xvec @ y \# yvec)) \langle N' \rangle$

$= iM(\nu * zvec) \langle N \rangle \implies$

Prop P'

and *rScope*: $\bigwedge P'. [\Psi \triangleright P \mapsto_{iM} (\nu * zvec) \langle N \rangle \prec P'] \implies \text{Prop } ((\nu x)P')$

shows *Prop P'*
 ⟨*proof*⟩

lemma *brOutputFreshSubject*:

fixes *x*::*name*
assumes $\Psi \triangleright P \mapsto_{iM} (\nu * xvec) \langle N \rangle \prec P'$
and $xvec \#* M$
and $x \# P$
shows $x \# M$
 ⟨*proof*⟩

lemma *brInputFreshSubject*:

fixes *x*::*name*
assumes $\Psi \triangleright P \mapsto_{iM} \langle N \rangle \prec P'$
and $x \# P$
shows $x \# M$
 ⟨*proof*⟩

lemma *resComm*:

fixes Ψ :: 'b

and $x :: name$
and $y :: name$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$
and $P :: ('a, 'b, 'c) psi$

assumes $x \# \Psi$
and $y \# \Psi$
and $eqvt Rel$
and $R1: \bigwedge \Psi' Q. (\Psi', Q, Q) \in Rel$
and $R2: \bigwedge \Psi' a b Q. \llbracket a \# \Psi'; b \# \Psi' \rrbracket \implies (\Psi', (\nu a)((\nu b)Q), (\nu b)((\nu a)Q)) \in Rel$
and $R3: \bigwedge \Psi' xvec yvec Q. \llbracket xvec \#* \Psi'; mset xvec = mset yvec \rrbracket \implies (\Psi', (\nu*xvec)Q, (\nu*yvec)Q) \in Rel$

shows $\Psi \triangleright (\nu x)((\nu y)P) \rightsquigarrow[Rel] (\nu y)((\nu x)P)$
<proof>

lemma *parAssocLeft*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $eqvt Rel$
and $C1: \bigwedge \Psi' S T U. (\Psi, (S \parallel T) \parallel U, S \parallel (T \parallel U)) \in Rel$
and $C2: \bigwedge xvec \Psi' S T U. \llbracket xvec \#* \Psi'; xvec \#* S \rrbracket \implies (\Psi', (\nu*xvec)((S \parallel T) \parallel U), S \parallel ((\nu*xvec)(T \parallel U))) \in Rel$
and $C3: \bigwedge xvec \Psi' S T U. \llbracket xvec \#* \Psi'; xvec \#* U \rrbracket \implies (\Psi', ((\nu*xvec)(S \parallel T)) \parallel U, (\nu*xvec)(S \parallel (T \parallel U))) \in Rel$
and $C4: \bigwedge \Psi' S T xvec. \llbracket (\Psi', S, T) \in Rel; xvec \#* \Psi' \rrbracket \implies (\Psi', (\nu*xvec)S, (\nu*xvec)T) \in Rel$

shows $\Psi \triangleright (P \parallel Q) \parallel R \rightsquigarrow[Rel] P \parallel (Q \parallel R)$
<proof>

lemma *parNilLeft*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $eqvt Rel$
and $C1: \bigwedge Q. (\Psi, Q \parallel \mathbf{0}, Q) \in Rel$

shows $\Psi \triangleright (P \parallel \mathbf{0}) \rightsquigarrow[Rel] P$
<proof>

lemma *parNilRight*:
fixes $\Psi :: 'b$

and $P :: ('a, 'b, 'c) \text{ psi}$
and $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

assumes $\text{eqvt } Rel$
and $C1: \bigwedge Q. (\Psi, Q, (Q \parallel \mathbf{0})) \in Rel$

shows $\Psi \triangleright P \rightsquigarrow[Rel] (P \parallel \mathbf{0})$
 $\langle \text{proof} \rangle$

lemma resNilLeft :
fixes $x :: \text{name}$
and $\Psi :: 'b$
and $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

shows $\Psi \triangleright (\nu x)\mathbf{0} \rightsquigarrow[Rel] \mathbf{0}$
 $\langle \text{proof} \rangle$

lemma resNilRight :
fixes $x :: \text{name}$
and $\Psi :: 'b$
and $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$

shows $\Psi \triangleright \mathbf{0} \rightsquigarrow[Rel] (\nu x)\mathbf{0}$
 $\langle \text{proof} \rangle$

lemma inputPushResLeft :
fixes $\Psi :: 'b$
and $x :: \text{name}$
and $M :: 'a$
and $xvec :: \text{name list}$
and $N :: 'a$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $\text{eqvt } Rel$
and $x \# \Psi$
and $x \# M$
and $x \# xvec$
and $x \# N$
and $C1: \bigwedge Q. (\Psi, Q, Q) \in Rel$

shows $\Psi \triangleright (\nu x)(M(\lambda * xvec N).P) \rightsquigarrow[Rel] M(\lambda * xvec N).(\nu x)P$
 $\langle \text{proof} \rangle$

lemma inputPushResRight :
fixes $\Psi :: 'b$
and $x :: \text{name}$
and $M :: 'a$
and $xvec :: \text{name list}$
and $N :: 'a$

and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $\text{eqvt } Rel$

and $x \# \Psi$

and $x \# M$

and $x \# \text{xvec}$

and $x \# N$

and $C1: \bigwedge Q. (\Psi, Q, Q) \in Rel$

shows $\Psi \triangleright M(\lambda * \text{xvec } N).(\nu x)P \rightsquigarrow[Rel] (\nu x)(M(\lambda * \text{xvec } N).P)$

<proof>

lemma *outputPushResLeft*:

fixes $\Psi :: 'b$

and $x :: \text{name}$

and $M :: 'a$

and $N :: 'a$

and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $\text{eqvt } Rel$

and $x \# \Psi$

and $x \# M$

and $x \# N$

and $C1: \bigwedge Q. (\Psi, Q, Q) \in Rel$

shows $\Psi \triangleright (\nu x)(M\langle N \rangle.P) \rightsquigarrow[Rel] M\langle N \rangle.(\nu x)P$

<proof>

lemma *broutputNoBind*:

fixes $\Psi :: 'b$

and $M :: 'a$

and $N :: 'a$

and $P :: ('a, 'b, 'c) \text{ psi}$

and $\alpha :: 'a \text{ action}$

and $P' :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \triangleright M\langle N \rangle.P \mapsto (\text{jK}(\nu * \text{xvec})\langle N' \rangle) \prec P'$

shows $\text{xvec} = []$

<proof>

lemma *broutputObjectEq*:

fixes $\Psi :: 'b$

and $M :: 'a$

and $N :: 'a$

and $P :: ('a, 'b, 'c) \text{ psi}$

and $\alpha :: 'a \text{ action}$

and $P' :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \triangleright M\langle N \rangle.P \mapsto (\text{jK}(\nu * \text{xvec})\langle N' \rangle) \prec P'$

shows $N = N'$
 $\langle proof \rangle$

lemma *brOutputOutputCases*[*consumes 1, case-names cBrOutput*]:

fixes $\Psi :: 'b$
and $M :: 'a$
and $N :: 'a$
and $P :: ('a, 'b, 'c) psi$
and $\alpha :: 'a action$
and $P' :: ('a, 'b, 'c) psi$

assumes *Trans*: $\Psi \triangleright M \langle N \rangle . P \mapsto (jK(\nu * xvec) \langle N' \rangle) \prec P'$
and *rBrOutput*: $\bigwedge K. \Psi \vdash M \preceq K \implies Prop (jK \langle N \rangle) P$

shows $Prop (jK(\nu * xvec) \langle N' \rangle) P'$
 $\langle proof \rangle$

lemma *outputPushResRight*:

fixes $\Psi :: 'b$
and $x :: name$
and $M :: 'a$
and $N :: 'a$
and $P :: ('a, 'b, 'c) psi$

assumes *eqvt Rel*
and $x \# \Psi$
and $x \# M$
and $x \# N$
and *C1*: $\bigwedge Q. (\Psi, Q, Q) \in Rel$

shows $\Psi \triangleright M \langle N \rangle . (\nu x) P \rightsquigarrow [Rel] (\nu x) (M \langle N \rangle . P)$
 $\langle proof \rangle$

lemma *casePushResLeft*:

fixes $\Psi :: 'b$
and $x :: name$
and $Cs :: ('c \times ('a, 'b, 'c) psi) list$

assumes *eqvt Rel*
and $x \# \Psi$
and $x \# map fst Cs$
and *C1*: $\bigwedge Q. (\Psi, Q, Q) \in Rel$

shows $\Psi \triangleright (\nu x) (Cases Cs) \rightsquigarrow [Rel] Cases (map (\lambda(\varphi, P). (\varphi, (\nu x) P)) Cs)$
 $\langle proof \rangle$

lemma *casePushResRight*:

fixes $\Psi :: 'b$
and $x :: name$

and $Cs :: ('c \times ('a, 'b, 'c) \text{psi}) \text{list}$

assumes $\text{eqvt } Rel$

and $x \# \Psi$

and $x \# \text{map } \text{fst } Cs$

and $C1: \bigwedge Q. (\Psi, Q, Q) \in Rel$

shows $\Psi \triangleright \text{Cases } (\text{map } (\lambda(\varphi, P). (\varphi, (\nu x)P)) Cs) \rightsquigarrow[Rel] (\nu x)(\text{Cases } Cs)$

 $\langle \text{proof} \rangle$

lemma $\text{resInputCases}[\text{consumes } 5, \text{case-names } cRes]:$

fixes $\Psi :: 'b$

and $x :: \text{name}$

and $P :: ('a, 'b, 'c) \text{psi}$

and $M :: 'a$

and $N :: 'a$

and $P' :: ('a, 'b, 'c) \text{psi}$

and $C :: 'd::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright (\nu x)P \mapsto M(N) \prec P'$

and $x \# \Psi$

and $x \# M$

and $x \# N$

and $x \# P'$

and $rScope: \bigwedge P'. [\Psi \triangleright P \mapsto M(N) \prec P'] \implies \text{Prop } ((\nu x)P')$

shows $\text{Prop } P'$

 $\langle \text{proof} \rangle$

lemma $\text{resBrInputCases}[\text{consumes } 5, \text{case-names } cRes]:$

fixes $\Psi :: 'b$

and $x :: \text{name}$

and $P :: ('a, 'b, 'c) \text{psi}$

and $M :: 'a$

and $N :: 'a$

and $P' :: ('a, 'b, 'c) \text{psi}$

and $C :: 'd::\text{fs-name}$

assumes $\text{Trans}: \Psi \triangleright (\nu x)P \mapsto_i M(N) \prec P'$

and $x \# \Psi$

and $x \# M$

and $x \# N$

and $x \# P'$

and $rScope: \bigwedge P'. [\Psi \triangleright P \mapsto_i M(N) \prec P'] \implies \text{Prop } ((\nu x)P')$

shows $\text{Prop } P'$

 $\langle \text{proof} \rangle$

lemma $\text{swap-supp}:$


```

fixes  $x :: name$ 
  and  $y :: name$ 
  and  $N :: 'a$ 

assumes  $y \in \text{supp } N$ 

shows  $x \in \text{supp } ([x, y] \cdot N)$ 
   $\langle \text{proof} \rangle$ 

lemma swap-supp':
  fixes  $x :: name$ 
  and  $y :: name$ 
  and  $N :: 'a$ 

assumes  $x \in \text{supp } N$ 

shows  $y \in \text{supp } ([x, y] \cdot N)$ 
   $\langle \text{proof} \rangle$ 

lemma brOutputFreshSubjectChain:
  fixes  $\Psi :: 'b$ 
  and  $Q :: ('a, 'b, 'c) \text{ psi}$ 
  and  $M :: 'a$ 
  and  $xvec :: name \text{ list}$ 
  and  $N :: 'a$ 
  and  $Q' :: ('a, 'b, 'c) \text{ psi}$ 
  and  $yvec :: name \text{ list}$ 

assumes  $\Psi \triangleright Q \mapsto \text{!}M(\nu*xvec)\langle N \rangle \prec Q'$ 
  and  $xvec \#* M$ 
  and  $yvec \#* Q$ 

shows  $yvec \#* M$ 
   $\langle \text{proof} \rangle$ 

lemma scopeExtLeft:
  fixes  $x :: name$ 
  and  $P :: ('a, 'b, 'c) \text{ psi}$ 
  and  $\Psi :: 'b$ 
  and  $Q :: ('a, 'b, 'c) \text{ psi}$ 
  and  $Rel :: ('b \times ('a, 'b, 'c) \text{ psi} \times ('a, 'b, 'c) \text{ psi}) \text{ set}$ 

assumes  $x \# P$ 
  and  $x \# \Psi$ 
  and  $\text{eqvt } Rel$ 
  and  $C1: \bigwedge \Psi' R. (\Psi', R, R) \in Rel$ 
  and  $C2: \bigwedge y \Psi' R S \text{ zvec}. \llbracket y \# \Psi'; y \# R; \text{zvec} \#* \Psi \rrbracket \implies (\Psi', (\nu y)((\nu*\text{zvec})(R$ 
   $\parallel S)), (\nu*\text{zvec})(R \parallel (\nu y)S)) \in Rel$ 
  and  $C3: \bigwedge \Psi' \text{zvec } R y. \llbracket y \# \Psi'; \text{zvec} \#* \Psi \rrbracket \implies (\Psi', (\nu y)((\nu*\text{zvec})R),$ 

```

$(\nu^*zvec)(\nu y)R) \in Rel$
 — Addition for Broadcast
and $C_4: \bigwedge \Psi' R S zvec. \llbracket zvec \#* R; zvec \#* \Psi' \rrbracket \implies (\Psi', ((\nu^*zvec)(R \parallel S)), (R \parallel ((\nu^*zvec)S))) \in Rel$

shows $\Psi \triangleright (\nu x)(P \parallel Q) \rightsquigarrow[Rel] P \parallel (\nu x)Q$
 $\langle proof \rangle$

lemma *scopeExtRight*:

fixes $x :: name$
and $P :: ('a, 'b, 'c) psi$
and $\Psi :: 'b$
and $Q :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes $x \# P$

and $x \# \Psi$
and *eqvt* Rel
and $C1: \bigwedge \Psi' R. (\Psi, R, R) \in Rel$
and $C2: \bigwedge y \Psi' R S zvec. \llbracket y \# \Psi'; y \# R; zvec \#* \Psi' \rrbracket \implies (\Psi', (\nu^*zvec)(R \parallel (\nu y)S), (\nu y)((\nu^*zvec)(R \parallel S))) \in Rel$
 — Addition for Broadcast
and $C3: \bigwedge \Psi' R S zvec. \llbracket zvec \#* R; zvec \#* \Psi' \rrbracket \implies (\Psi', (R \parallel ((\nu^*zvec)S)), ((\nu^*zvec)(R \parallel S))) \in Rel$

shows $\Psi \triangleright P \parallel (\nu x)Q \rightsquigarrow[Rel] (\nu x)(P \parallel Q)$
 $\langle proof \rangle$

lemma *simParComm*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $Rel :: ('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

assumes *eqvt* Rel

and $C1: \bigwedge \Psi' R S. (\Psi', R \parallel S, S \parallel R) \in Rel$
and $C2: \bigwedge \Psi' R S xvec. \llbracket (\Psi', R, S) \in Rel; xvec \#* \Psi' \rrbracket \implies (\Psi', (\nu^*xvec)R, (\nu^*xvec)S) \in Rel$

shows $\Psi \triangleright P \parallel Q \rightsquigarrow[Rel] Q \parallel P$
 $\langle proof \rangle$

lemma *bangExtLeft*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$

assumes *guarded* P

and $\bigwedge \Psi' Q. (\Psi', Q, Q) \in Rel$

shows $\Psi \triangleright !P \rightsquigarrow[Rel] P \parallel !P$
 ⟨proof⟩

lemma *bangExtRight*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$

assumes $C1: \bigwedge \Psi' Q. (\Psi', Q, Q) \in Rel$

shows $\Psi \triangleright P \parallel !P \rightsquigarrow[Rel] !P$
 ⟨proof⟩

end

end

theory *Bisim-Pres*

imports *Bisimulation Sim-Pres*

begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Bisim_Pres* from [1].

context *env* **begin**

lemma *bisimInputPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $xvec :: name list$
and $N :: 'a$

assumes $\bigwedge Tvec. length\ xvec = length\ Tvec \implies \Psi \triangleright P[xvec ::= Tvec] \sim Q[xvec ::= Tvec]$

shows $\Psi \triangleright M(\lambda * xvec\ N).P \sim M(\lambda * xvec\ N).Q$
 ⟨proof⟩

lemma *bisimOutputPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $M :: 'a$
and $N :: 'a$

assumes $\Psi \triangleright P \sim Q$

shows $\Psi \triangleright M\langle N \rangle.P \sim M\langle N \rangle.Q$
 ⟨proof⟩

lemma *bisimCasePres*:

fixes $\Psi :: 'b$
and $CsP :: ('c \times ('a, 'b, 'c) psi) list$
and $CsQ :: ('c \times ('a, 'b, 'c) psi) list$

assumes $\bigwedge \varphi P. (\varphi, P) \in set\ CsP \implies \exists Q. (\varphi, Q) \in set\ CsQ \wedge guarded\ Q \wedge \Psi$
 $\triangleright P \sim Q$
and $\bigwedge \varphi Q. (\varphi, Q) \in set\ CsQ \implies \exists P. (\varphi, P) \in set\ CsP \wedge guarded\ P \wedge \Psi \triangleright$
 $P \sim Q$

shows $\Psi \triangleright Cases\ CsP \sim Cases\ CsQ$
 $\langle proof \rangle$

lemma *bisimResPres*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $x :: name$

assumes $\Psi \triangleright P \sim Q$
and $x \# \Psi$

shows $\Psi \triangleright (\nu x)P \sim (\nu x)Q$
 $\langle proof \rangle$

lemma *bisimResChainPres*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $xvec :: name\ list$

assumes $\Psi \triangleright P \sim Q$
and $xvec \#* \Psi$

shows $\Psi \triangleright (\nu *xvec)P \sim (\nu *xvec)Q$
 $\langle proof \rangle$

lemma *bisimParPresAux*:
fixes $\Psi :: 'b$
and $\Psi_R :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$
and $A_R :: name\ list$

assumes $\Psi \otimes \Psi_R \triangleright P \sim Q$
and $FrR: extractFrame\ R = \langle A_R, \Psi_R \rangle$
and $A_R \#* \Psi$
and $A_R \#* P$
and $A_R \#* Q$

shows $\Psi \triangleright P \parallel R \sim Q \parallel R$
 $\langle proof \rangle$

lemma *bisimParPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

assumes $\Psi \triangleright P \sim Q$

shows $\Psi \triangleright P \parallel R \sim Q \parallel R$
 $\langle proof \rangle$

end

end

theory *Bisim-Struct-Cong*

imports *Bisim-Pres Sim-Struct-Cong Psi-Calculi.Structural-Congruence*

begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Bisim_Struct_Cong* from [1].

context *env* **begin**

lemma *bisimParComm*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$

shows $\Psi \triangleright P \parallel Q \sim Q \parallel P$
 $\langle proof \rangle$

inductive-set *resCommRel* :: $('b \times ('a, 'b, 'c) psi \times ('a, 'b, 'c) psi) set$

where

resCommRel-refl : $(\Psi, P, P) \in resCommRel$

| *resCommRel-swap* : $\llbracket a \# \Psi; b \# \Psi \rrbracket \implies (\Psi, (\nu a)(\nu b)P, (\nu b)(\nu a)P) \in resCommRel$

| *resCommRel-res* : $\llbracket (\Psi, P, Q) \in resCommRel; a \# \Psi \rrbracket \implies (\Psi, (\nu a)P, (\nu a)Q) \in resCommRel$

lemma *eqvtResCommRel*: *eqvt resCommRel*

$\langle proof \rangle$

lemma *resCommRelStarRes*:

assumes $(\Psi, P, Q) \in resCommRel^*$

and $a \# \Psi$

shows $(\Psi, (\nu a)P, (\nu a)Q) \in resCommRel^*$

<proof>

lemma *resCommRelStarResChain*:

assumes $(\Psi, P, Q) \in \text{resCommRel}^*$

and $xvec \#* \Psi$

shows $(\Psi, (\nu * xvec) P, (\nu * xvec) Q) \in \text{resCommRel}^*$

<proof>

lemma *length-induct1* [*consumes 0, case-names Nil Cons*]:

assumes $b: P \square$

and $s: \bigwedge x \ xvec. [\bigwedge yvec. \text{length } xvec = \text{length } yvec \implies P \ yvec] \implies P(x \# xvec)$

shows $P \ xvec$

<proof>

lemma *oneStepPerm-in-rel*:

assumes $xvec \#* \Psi$

shows $(\Psi, (\nu * xvec) P, (\nu * (\text{rotate1 } xvec)) P) \in \text{resCommRel}^*$

<proof>

lemma *fresh-same-multiset*:

fixes $xvec :: \text{name list}$

and $yvec :: \text{name list}$

assumes $\text{mset } xvec = \text{mset } yvec$

and $xvec \#* X$

shows $yvec \#* X$

<proof>

lemma *nStepPerm-in-rel*:

assumes $xvec \#* \Psi$

shows $(\Psi, (\nu * xvec) P, (\nu * (\text{rotate } n \ xvec)) P) \in \text{resCommRel}^*$

<proof>

lemma *any-perm-in-rel*:

assumes $xvec \#* \Psi$

and $\text{mset } xvec = \text{mset } yvec$

shows $(\Psi, (\nu * xvec) P, (\nu * yvec) P) \in \text{resCommRel}^*$

<proof>

lemma *bisimResComm*:

fixes $x :: \text{name}$

and $\Psi :: 'b$

and $y :: \text{name}$

and $P :: ('a, 'b, 'c) \text{psi}$

shows $\Psi \triangleright (\nu x)((\nu y)P) \sim (\nu y)((\nu x)P)$

<proof>

lemma *bisimResComm'*:

fixes $x \quad :: \text{name}$

```

and  $\Psi$   :: 'b
and  $xvec$  :: name list
and  $P$     :: ('a, 'b, 'c) psi

assumes  $x \# \Psi$ 
and  $xvec \#* \Psi$ 

shows  $\Psi \triangleright (\nu x)((\nu *xvec)P) \sim (\nu *xvec)((\nu x)P)$ 
  <proof>

lemma bisimParPresSym:
  fixes  $\Psi$  :: 'b
    and  $P$  :: ('a, 'b, 'c) psi
    and  $Q$  :: ('a, 'b, 'c) psi
    and  $R$  :: ('a, 'b, 'c) psi

assumes  $\Psi \triangleright P \sim Q$ 

shows  $\Psi \triangleright R \parallel P \sim R \parallel Q$ 
  <proof>

lemma bisimScopeExt:
  fixes  $x$  :: name
    and  $\Psi$  :: 'b
    and  $P$  :: ('a, 'b, 'c) psi
    and  $Q$  :: ('a, 'b, 'c) psi

assumes  $x \# P$ 

shows  $\Psi \triangleright (\nu x)(P \parallel Q) \sim P \parallel (\nu x)Q$ 
  <proof>

lemma bisimScopeExtChain:
  fixes  $xvec$  :: name list
    and  $\Psi$     :: 'b
    and  $P$     :: ('a, 'b, 'c) psi
    and  $Q$     :: ('a, 'b, 'c) psi

assumes  $xvec \#* \Psi$ 
and  $xvec \#* P$ 

shows  $\Psi \triangleright (\nu *xvec)(P \parallel Q) \sim P \parallel ((\nu *xvec)Q)$ 
  <proof>

lemma bisimParAssoc:
  fixes  $\Psi$  :: 'b
    and  $P$  :: ('a, 'b, 'c) psi
    and  $Q$  :: ('a, 'b, 'c) psi
    and  $R$  :: ('a, 'b, 'c) psi

```

shows $\Psi \triangleright (P \parallel Q) \parallel R \sim P \parallel (Q \parallel R)$
<proof>

lemma *bisimParNil*:
fixes $P :: ('a, 'b, 'c) \text{ psi}$

shows $\Psi \triangleright P \parallel \mathbf{0} \sim P$
<proof>

lemma *bisimResNil*:
fixes $x :: \text{ name}$
and $\Psi :: 'b$

shows $\Psi \triangleright (\nu x)\mathbf{0} \sim \mathbf{0}$
<proof>

lemma *bisimOutputPushRes*:
fixes $x :: \text{ name}$
and $\Psi :: 'b$
and $M :: 'a$
and $N :: 'a$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $x \# M$
and $x \# N$

shows $\Psi \triangleright (\nu x)(M\langle N \rangle.P) \sim M\langle N \rangle.(\nu x)P$
<proof>

lemma *bisimInputPushRes*:
fixes $x :: \text{ name}$
and $\Psi :: 'b$
and $M :: 'a$
and $xvec :: \text{ name list}$
and $N :: 'a$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $x \# M$
and $x \# xvec$
and $x \# N$

shows $\Psi \triangleright (\nu x)(M(\lambda * xvec N).P) \sim M(\lambda * xvec N).(\nu x)P$
<proof>

lemma *bisimCasePushRes*:
fixes $x :: \text{ name}$
and $\Psi :: 'b$
and $Cs :: ('c \times ('a, 'b, 'c) \text{ psi}) \text{ list}$

assumes $x \# (\text{map fst } Cs)$

shows $\Psi \triangleright (\nu x)(\text{Cases } Cs) \sim \text{Cases}(\text{map } (\lambda(\varphi, P). (\varphi, (\nu x)P)) Cs)$
 $\langle \text{proof} \rangle$

lemma *bangExt*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes *guarded* P

shows $\Psi \triangleright !P \sim P \parallel !P$
 $\langle \text{proof} \rangle$

lemma *bisimScopeExtSym*:
fixes $x :: \text{name}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $x \# \Psi$
and $x \# Q$

shows $\Psi \triangleright (\nu x)(P \parallel Q) \sim ((\nu x)P) \parallel Q$
 $\langle \text{proof} \rangle$

lemma *bisimScopeExtChainSym*:
fixes $xvec :: \text{name list}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $P :: ('a, 'b, 'c) \text{ psi}$

assumes $xvec \#* \Psi$
and $xvec \#* Q$

shows $\Psi \triangleright (\nu *xvec)(P \parallel Q) \sim ((\nu *xvec)P) \parallel Q$
 $\langle \text{proof} \rangle$

lemma *bisimParPresAuxSym*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $R :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \otimes \Psi_R \triangleright P \sim Q$
and $\text{extractFrame } R = \langle A_R, \Psi_R \rangle$
and $A_R \#* \Psi$
and $A_R \#* P$
and $A_R \#* Q$

shows $\Psi \triangleright R \parallel P \sim R \parallel Q$
 ⟨*proof*⟩

lemma *bangDerivative*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $\alpha :: 'a \text{ action}$
and $P' :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \triangleright !P \xrightarrow{\alpha} \prec P'$
and $\Psi \triangleright P \sim Q$
and $bn \ \alpha \ \#* \ \Psi$
and $bn \ \alpha \ \#* \ P$
and $bn \ \alpha \ \#* \ Q$
and $bn \ \alpha \ \#* \ \text{subject } \alpha$
and *guarded* Q

obtains $Q' R T$ **where** $\Psi \triangleright !Q \xrightarrow{\alpha} \prec Q'$ **and** $\Psi \triangleright P' \sim R \parallel !P$ **and** $\Psi \triangleright Q' \sim T \parallel !Q$ **and** $\Psi \triangleright R \sim T$
and $((\text{supp } R)::\text{name set}) \subseteq \text{supp } P'$ **and** $((\text{supp } T)::\text{name set}) \subseteq \text{supp } Q'$
 ⟨*proof*⟩

lemma *structCongBisim*:

fixes $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$

assumes $P \equiv_s Q$

shows $P \sim Q$
 ⟨*proof*⟩

lemma *bisimBangPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \triangleright P \sim Q$
and *guarded* P
and *guarded* Q

shows $\Psi \triangleright !P \sim !Q$
 ⟨*proof*⟩

end

end

theory *Bisim-Subst*

imports *Bisim-Struct-Cong Psi-Calculi.Close-Subst*
begin

This file is a (heavily modified) variant of the theory *Psi_Calculi.Bisim_Subst* from [1].

context *env* **begin**

abbreviation

bisimSubstJudge ($- \triangleright - \sim_s -$ [70, 70, 70] 65) **where** $\Psi \triangleright P \sim_s Q \equiv (\Psi, P, Q) \in \text{closeSubst } \text{bisim}$

abbreviation

bisimSubstNilJudge ($- \sim_s -$ [70, 70] 65) **where** $P \sim_s Q \equiv \text{SBottom}' \triangleright P \sim_s Q$

lemmas *bisimSubstClosed*[*eqvt*] = *closeSubstClosed*[*OF bisimEqvt*]

lemmas *bisimSubstEqvt*[*simp*] = *closeSubstEqvt*[*OF bisimEqvt*]

lemma *bisimSubstOutputPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and $N :: 'a$

assumes $\Psi \triangleright P \sim_s Q$

shows $\Psi \triangleright M\langle N \rangle.P \sim_s M\langle N \rangle.Q$
<proof>

lemma *seqSubstInputChain*[*simp*]:

fixes *xvec* :: *name list*
and $N :: 'a$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $\sigma :: (\text{name list} \times 'a \text{ list}) \text{ list}$

assumes *xvec* $\#^* \sigma$

shows *seqSubs'* (*inputChain* *xvec* $N P$) $\sigma = \text{inputChain } \text{xvec } (\text{substTerm.seqSubst } N \sigma) (\text{seqSubs } P \sigma)$
<proof>

lemma *bisimSubstInputPres*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $M :: 'a$
and *xvec* :: *name list*
and $N :: 'a$

assumes $\Psi \triangleright P \sim_s Q$

and *xvec* $\#^* \Psi$
and *distinct* *xvec*

shows $\Psi \triangleright M(\lambda*xvec N).P \sim_s M(\lambda*xvec N).Q$
 $\langle proof \rangle$

lemma *bisimSubstCasePresAux*:

fixes $\Psi :: 'b$
and $CsP :: ('c \times ('a, 'b, 'c) psi) list$
and $CsQ :: ('c \times ('a, 'b, 'c) psi) list$

assumes $C1: \bigwedge \varphi P. (\varphi, P) \in set CsP \implies \exists Q. (\varphi, Q) \in set CsQ \wedge guarded Q \wedge \Psi \triangleright P \sim_s Q$
and $C2: \bigwedge \varphi Q. (\varphi, Q) \in set CsQ \implies \exists P. (\varphi, P) \in set CsP \wedge guarded P \wedge \Psi \triangleright P \sim_s Q$

shows $\Psi \triangleright Cases CsP \sim_s Cases CsQ$
 $\langle proof \rangle$

lemma *bisimSubstReflexive*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$

shows $\Psi \triangleright P \sim_s P$
 $\langle proof \rangle$

lemma *bisimSubstTransitive*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

assumes $\Psi \triangleright P \sim_s Q$
and $\Psi \triangleright Q \sim_s R$

shows $\Psi \triangleright P \sim_s R$
 $\langle proof \rangle$

lemma *bisimSubstSymmetric*:

fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$

assumes $\Psi \triangleright P \sim_s Q$

shows $\Psi \triangleright Q \sim_s P$
 $\langle proof \rangle$

lemma *bisimSubstCasePres*:

fixes $\Psi :: 'b$
and $CsP :: ('c \times ('a, 'b, 'c) psi) list$

and $CsQ :: ('c \times ('a, 'b, 'c) \text{ psi}) \text{ list}$

assumes $\text{length } CsP = \text{length } CsQ$
and $C: \bigwedge(i::\text{nat}) \varphi P \varphi' Q. \llbracket i \leq \text{length } CsP; (\varphi, P) = \text{nth } CsP \ i; (\varphi', Q) = \text{nth } CsQ \ i \rrbracket \implies \varphi = \varphi' \wedge \Psi \triangleright P \sim_s Q \wedge \text{guarded } P \wedge \text{guarded } Q$

shows $\Psi \triangleright \text{Cases } CsP \sim_s \text{Cases } CsQ$
<proof>

lemma *bisimSubstParPres*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $R :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \triangleright P \sim_s Q$

shows $\Psi \triangleright P \parallel R \sim_s Q \parallel R$
<proof>

lemma *bisimSubstResPres*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$
and $x :: \text{name}$

assumes $\Psi \triangleright P \sim_s Q$
and $x \# \Psi$

shows $\Psi \triangleright (\nu x)P \sim_s (\nu x)Q$
<proof>

lemma *bisimSubstBangPres*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) \text{ psi}$
and $Q :: ('a, 'b, 'c) \text{ psi}$

assumes $\Psi \triangleright P \sim_s Q$
and *guarded* P
and *guarded* Q

shows $\Psi \triangleright !P \sim_s !Q$
<proof>

lemma *substNil[simp]*:
fixes $xvec :: \text{name list}$
and $Tvec :: 'a \text{ list}$

assumes *wellFormedSubst* σ

and *distinct xvec*

shows $\mathbf{0}[\langle\sigma\rangle] = \mathbf{0}$
<proof>

lemma *bisimSubstParNil*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$

shows $\Psi \triangleright P \parallel \mathbf{0} \sim_s P$
<proof>

lemma *bisimSubstParComm*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$

shows $\Psi \triangleright P \parallel Q \sim_s Q \parallel P$
<proof>

lemma *bisimSubstParAssoc*:
fixes $\Psi :: 'b$
and $P :: ('a, 'b, 'c) psi$
and $Q :: ('a, 'b, 'c) psi$
and $R :: ('a, 'b, 'c) psi$

shows $\Psi \triangleright (P \parallel Q) \parallel R \sim_s P \parallel (Q \parallel R)$
<proof>

lemma *bisimSubstResNil*:
fixes $\Psi :: 'b$
and $x :: name$

shows $\Psi \triangleright (\nu x)\mathbf{0} \sim_s \mathbf{0}$
<proof>

lemma *seqSubst2*:
fixes $x :: name$
and $P :: ('a, 'b, 'c) psi$

assumes *wellFormedSubst* σ
and $x \# \sigma$
and $x \# P$

shows $x \# P[\langle\sigma\rangle]$
<proof>

notation *substTerm.seqSubst* $(-\langle-\rangle)$ [100, 100] 100)

lemma *bisimSubstScopeExt*:

fixes $\Psi :: 'b$
and $x :: \text{name}$
and $P :: ('a, 'b, 'c) \text{psi}$
and $Q :: ('a, 'b, 'c) \text{psi}$

assumes $x \# P$

shows $\Psi \triangleright (\nu x)(P \parallel Q) \sim_s P \parallel (\nu x)Q$
<proof>

lemma *bisimSubstCasePushRes*:

fixes $x :: \text{name}$
and $\Psi :: 'b$
and $Cs :: ('c \times ('a, 'b, 'c) \text{psi}) \text{list}$

assumes $x \# \text{map fst } Cs$

shows $\Psi \triangleright (\nu x)(\text{Cases } Cs) \sim_s \text{Cases map } (\lambda(\varphi, P). (\varphi, (\nu x)P)) Cs$
<proof>

lemma *bisimSubstOutputPushRes*:

fixes $x :: \text{name}$
and $\Psi :: 'b$
and $M :: 'a$
and $N :: 'a$
and $P :: ('a, 'b, 'c) \text{psi}$

assumes $x \# M$
and $x \# N$

shows $\Psi \triangleright (\nu x)(M \langle N \rangle . P) \sim_s M \langle N \rangle . (\nu x)P$
<proof>

lemma *bisimSubstInputPushRes*:

fixes $x :: \text{name}$
and $\Psi :: 'b$
and $M :: 'a$
and $xvec :: \text{name list}$
and $N :: 'a$

assumes $x \# M$
and $x \# xvec$
and $x \# N$

shows $\Psi \triangleright (\nu x)(M(\lambda * xvec N).P) \sim_s M(\lambda * xvec N).(\nu x)P$
<proof>

lemma *bisimSubstResComm*:

```

fixes  $x :: name$ 
and  $y :: name$ 

shows  $\Psi \triangleright (\nu x)((\nu y)P) \sim_s (\nu y)((\nu x)P)$ 
 $\langle proof \rangle$ 

lemma bisimSubstExtBang:
fixes  $\Psi :: 'b$ 
and  $P :: ('a, 'b, 'c) psi$ 

assumes guarded P

shows  $\Psi \triangleright !P \sim_s P \parallel !P$ 
 $\langle proof \rangle$ 

lemma structCongBisimSubst:
fixes  $P :: ('a, 'b, 'c) psi$ 
and  $Q :: ('a, 'b, 'c) psi$ 

assumes  $P \equiv_s Q$ 

shows  $P \sim_s Q$ 
 $\langle proof \rangle$ 

end

end

theory Broadcast-Thms
imports Broadcast-Chain Broadcast-Frame Semantics Simulation Bisimulation
Sim-Pres
Sim-Struct-Cong Bisim-Pres Bisim-Struct-Cong Bisim-Subst
begin

context env
begin

```

2.1 Syntax

2.1.1 Psi calculus agents – Definition 2

M, N range over message terms. P, Q range over processes. C ranges over cases.

- Output: $M\langle N \rangle.P$
- Input: $M(\lambda *xvec N).P$
- Case: $Case C$
- Par: $P \parallel Q$

- Res: $(\nu x)P$
- Assert: $\{\Psi\}$
- Bang: $!P$

- Cases: $\square \varphi \Rightarrow P C$

2.1.2 Parameters – Definition 1

- Channel equivalence: $M \leftrightarrow N$
- Composition: $\Psi_P \otimes \Psi_Q$
- Unit: $\mathbf{1}$
- Entailment: $\Psi \vdash \varphi$

2.1.3 Extra predicates for broadcast – Definition 5

- Output connectivity: $M \preceq N$
- Input connectivity: $M \succeq N$

2.1.4 Transitions – Definition 3

- $\Psi \triangleright P \longmapsto \alpha P'$

2.1.5 Actions (α) – Definition 7

- Input: $M(N)$
- Output: $M(\nu * xvec)\langle N \rangle$
- Broadcast input: $!M(N)$
- Broadcast output: $!M(\nu * xvec)\langle N \rangle$
- Silent action: τ

2.2 Semantics

2.2.1 Basic Psi semantics – Table 1

- Theorem *Input*:

$$\begin{aligned} & \llbracket \Psi \vdash M \leftrightarrow K; \text{distinct } xvec; \text{set } xvec \subseteq \text{supp } N; \text{length } xvec = \text{length } Tvec \rrbracket \\ & \implies \Psi \triangleright M(\lambda * xvec \ N).P \mapsto K(\lambda N[xvec ::= Tvec]) \prec P[xvec ::= Tvec] \end{aligned}$$

- Theorem *Output*:

$$\Psi \vdash M \leftrightarrow K \implies \Psi \triangleright M\langle N \rangle.P \mapsto K\langle N \rangle \prec P$$

- Theorem *Case*:

$$\llbracket \Psi \triangleright P \mapsto Rs; (\varphi, P) \text{ mem } Cs; \Psi \vdash \varphi; \text{guarded } P \rrbracket \implies \Psi \triangleright \text{Cases } Cs \mapsto Rs$$

- Theorems *Par1* and *Par2*:

$$\begin{aligned} & \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \alpha \prec P'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \text{bn } \alpha \#* Q; A_Q \#* \\ & \Psi; \\ & A_Q \#* P; A_Q \#* \alpha \rrbracket \\ & \implies \Psi \triangleright P \parallel Q \mapsto \alpha \prec P' \parallel Q \end{aligned}$$

$$\begin{aligned} & \llbracket \Psi \otimes \Psi_P \triangleright Q \mapsto \alpha \prec Q'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \text{bn } \alpha \#* P; A_P \#* \\ & \Psi; \\ & A_P \#* Q; A_P \#* \alpha \rrbracket \\ & \implies \Psi \triangleright P \parallel Q \mapsto \alpha \prec P \parallel Q' \end{aligned}$$

- Theorems *Comm1* and *Comm2*:

$$\begin{aligned} & \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\lambda N) \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \\ & \Psi \otimes \Psi_P \triangleright Q \mapsto K(\nu * xvec)\langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \\ & \Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K; A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* M; A_P \#* \\ & A_Q; A_Q \#* \Psi; \\ & A_Q \#* P; A_Q \#* Q; A_Q \#* K; xvec \#* P \rrbracket \\ & \implies \Psi \triangleright P \parallel Q \mapsto \tau \prec (\nu * xvec)P' \parallel Q' \end{aligned}$$

$$\begin{aligned} & \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto M(\nu * xvec)\langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \\ & \Psi \otimes \Psi_P \triangleright Q \mapsto K(\lambda N) \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; \Psi \otimes \Psi_P \otimes \\ & \Psi_Q \vdash M \leftrightarrow K; \\ & A_P \#* \Psi; A_P \#* P; A_P \#* Q; A_P \#* M; A_P \#* A_Q; A_Q \#* \Psi; A_Q \#* P; A_Q \\ & \#* Q; \\ & A_Q \#* K; xvec \#* Q \rrbracket \\ & \implies \Psi \triangleright P \parallel Q \mapsto \tau \prec (\nu * xvec)P' \parallel Q' \end{aligned}$$

- Theorem *Open*:

$$\begin{aligned}
& \llbracket \Psi \triangleright P \mapsto M(\nu*(xvec \textcircled{\small @} yvec)) \langle N \rangle \prec P'; x \in \text{supp } N; x \# \Psi; x \# M; x \# \\
& \text{xvec}; \\
& x \# yvec \rrbracket \\
& \implies \Psi \triangleright (\nu x)P \mapsto M(\nu*(xvec \textcircled{\small @} x \# yvec)) \langle N \rangle \prec P'
\end{aligned}$$

- Theorem *Scope*:

$$\llbracket \Psi \triangleright P \mapsto \alpha \prec P'; x \# \Psi; x \# \alpha \rrbracket \implies \Psi \triangleright (\nu x)P \mapsto \alpha \prec (\nu x)P'$$

- Theorem *Bang*:

$$\llbracket \Psi \triangleright P \parallel !P \mapsto Rs; \text{guarded } P \rrbracket \implies \Psi \triangleright !P \mapsto Rs$$

2.2.2 Broadcast rules – Table 2

- Theorem *BrInput*:

$$\begin{aligned}
& \llbracket \Psi \vdash K \succeq M; \text{distinct } xvec; \text{set } xvec \subseteq \text{supp } N; \text{length } xvec = \text{length } Tvec \rrbracket \\
& \implies \Psi \triangleright M(\lambda * xvec \ N).P \mapsto \imath K \langle N[xvec ::= Tvec] \rangle \prec P[xvec ::= Tvec]
\end{aligned}$$

- Theorem *BrOutput*:

$$\Psi \vdash M \preceq K \implies \Psi \triangleright M \langle N \rangle . P \mapsto \imath K \langle N \rangle \prec P$$

- Theorem *BrMerge*:

$$\begin{aligned}
& \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \\
& \Psi \otimes \Psi_P \triangleright Q \mapsto \imath M \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; A_P \# * \Psi; \\
& A_P \# * P; \\
& A_P \# * Q; A_P \# * M; A_P \# * A_Q; A_Q \# * \Psi; A_Q \# * P; A_Q \# * Q; A_Q \# * M \rrbracket \\
& \implies \Psi \triangleright P \parallel Q \mapsto \imath M \langle N \rangle \prec P' \parallel Q'
\end{aligned}$$

- Theorems *BrComm1* and *BrComm2*:

$$\begin{aligned}
& \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \\
& \Psi \otimes \Psi_P \triangleright Q \mapsto \imath M(\nu * xvec) \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; A_P \\
& \# * \Psi; \\
& A_P \# * P; A_P \# * Q; A_P \# * M; A_P \# * A_Q; A_Q \# * \Psi; A_Q \# * P; A_Q \# * Q; A_Q \\
& \# * M; \\
& xvec \# * P \rrbracket \\
& \implies \Psi \triangleright P \parallel Q \mapsto \imath M(\nu * xvec) \langle N \rangle \prec P' \parallel Q'
\end{aligned}$$

$$\begin{aligned}
& \llbracket \Psi \otimes \Psi_Q \triangleright P \mapsto \imath M(\nu * xvec) \langle N \rangle \prec P'; \text{extractFrame } P = \langle A_P, \Psi_P \rangle; \\
& \Psi \otimes \Psi_P \triangleright Q \mapsto \imath M \langle N \rangle \prec Q'; \text{extractFrame } Q = \langle A_Q, \Psi_Q \rangle; A_P \# * \Psi; \\
& A_P \# * P; \\
& A_P \# * Q; A_P \# * M; A_P \# * A_Q; A_Q \# * \Psi; A_Q \# * P; A_Q \# * Q; A_Q \# * M; \\
& xvec \# * Q \rrbracket \\
& \implies \Psi \triangleright P \parallel Q \mapsto \imath M(\nu * xvec) \langle N \rangle \prec P' \parallel Q'
\end{aligned}$$

- Theorem *BrClose*:

$$\begin{aligned} & \llbracket \Psi \triangleright P \longmapsto \downarrow M(\nu^* xvec) \langle N \rangle \prec P'; x \in \text{supp } M; x \# \Psi \rrbracket \\ & \implies \Psi \triangleright (\nu x)P \longmapsto \tau \prec (\nu x)((\nu^* xvec)P') \end{aligned}$$

- Theorem *BrOpen*:

$$\begin{aligned} & \llbracket \Psi \triangleright P \longmapsto \downarrow M(\nu^*(xvec @ yvec)) \langle N \rangle \prec P'; x \in \text{supp } N; x \# \Psi; x \# M; x \# \\ & \quad xvec; \\ & \quad x \# yvec \rrbracket \\ & \implies \Psi \triangleright (\nu x)P \longmapsto \downarrow M(\nu^*(xvec @ x \# yvec)) \langle N \rangle \prec P' \end{aligned}$$

2.2.3 Requirements for broadcast – Definition 6

- Theorem *chanOutConSupp*:

$$\Psi \vdash M \preceq N \implies \text{supp } N \subseteq \text{supp } M$$

- Theorem *chanInConSupp*:

$$\Psi \vdash N \succeq M \implies \text{supp } N \subseteq \text{supp } M$$

2.2.4 Strong bisimulation – Definition 4

- Theorem *bisim.step*:

$$\begin{aligned} & \llbracket \text{insertAssertion } (\text{extractFrame } P) \Psi \simeq_F \text{insertAssertion } (\text{extractFrame } Q) \\ & \quad \Psi; \\ & \quad \Psi \triangleright P \rightsquigarrow[\text{bisim}] Q; \forall \Psi'. \Psi \otimes \Psi' \triangleright P \sim Q; \Psi \triangleright Q \sim P \rrbracket \\ & \implies \Psi \triangleright P \sim Q \end{aligned}$$

2.3 Theorems

2.3.1 Congruence properties of strong bisimulation – Theorem 8

- Theorem *bisimOutputPres*:

$$\Psi \triangleright P \sim Q \implies \Psi \triangleright M \langle N \rangle . P \sim M \langle N \rangle . Q$$

- Theorem *bisimInputPres*:

$$\begin{aligned} & (\bigwedge Tvec. \text{length } xvec = \text{length } Tvec \implies \Psi \triangleright P[xvec ::= Tvec] \sim Q[xvec ::= Tvec]) \\ & \implies \\ & \Psi \triangleright M(\lambda^* xvec N).P \sim M(\lambda^* xvec N).Q \end{aligned}$$

- Theorem *bisimCasePres*:

$$\begin{aligned} & \llbracket \bigwedge \varphi P. (\varphi, P) \text{ mem } CsP \implies \exists Q. (\varphi, Q) \text{ mem } CsQ \wedge \text{ guarded } Q \wedge \Psi \triangleright P \\ & \sim Q; \\ & \llbracket \bigwedge \varphi Q. (\varphi, Q) \text{ mem } CsQ \implies \exists P. (\varphi, P) \text{ mem } CsP \wedge \text{ guarded } P \wedge \Psi \triangleright P \\ & \sim Q \rrbracket \\ & \implies \Psi \triangleright \text{Cases } CsP \sim \text{Cases } CsQ \end{aligned}$$

- Theorems *bisimParPres* and *bisimParPresSym*:

$$\Psi \triangleright P \sim Q \implies \Psi \triangleright P \parallel R \sim Q \parallel R$$

$$\Psi \triangleright P \sim Q \implies \Psi \triangleright R \parallel P \sim R \parallel Q$$

- Theorem *bisimResPres*:

$$\llbracket \Psi \triangleright P \sim Q; x \# \Psi \rrbracket \implies \Psi \triangleright (\nu x)P \sim (\nu x)Q$$

- Theorem *bisimBangPres*:

$$\llbracket \Psi \triangleright P \sim Q; \text{ guarded } P; \text{ guarded } Q \rrbracket \implies \Psi \triangleright !P \sim !Q$$

2.3.2 Strong congruence, bisimulation closed under substitution – Definition 9

- Theorem *closeSubst_def*:

$$\begin{aligned} & \text{closeSubst Rel} \equiv \\ & \{(\Psi, P, Q) \mid \Psi P Q. \forall \sigma. \text{ wellFormedSubst } \sigma \longrightarrow (\Psi, P[\langle \sigma \rangle], Q[\langle \sigma \rangle]) \in \\ & \text{Rel}\} \end{aligned}$$

- $\Psi \triangleright P \sim_s Q$

2.3.3 Strong congruence is a process congruence for all Ψ – Theorem 10

- Theorem *bisimSubstOutputPres*:

$$\Psi \triangleright P \sim_s Q \implies \Psi \triangleright M\langle N \rangle.P \sim_s M\langle N \rangle.Q$$

- Theorem *bisimSubstInputPres*:

$$\llbracket \Psi \triangleright P \sim_s Q; \text{ xvec } \#^* \Psi; \text{ distinct xvec} \rrbracket \implies \Psi \triangleright M(\lambda^* \text{xvec } N).P \sim_s M(\lambda^* \text{xvec } N).Q$$

- Theorem *bisimSubstCasePres*:

$$\begin{aligned} & \llbracket \text{length } CsP = \text{length } CsQ; \\ & \bigwedge i \varphi P \varphi' Q. \\ & \quad \llbracket i \leq \text{length } CsP; (\varphi, P) = CsP ! i; (\varphi', Q) = CsQ ! i \rrbracket \\ & \quad \implies \varphi = \varphi' \wedge \Psi \triangleright P \sim_s Q \wedge \text{guarded } P \wedge \text{guarded } Q \rrbracket \\ & \implies \Psi \triangleright \text{Cases } CsP \sim_s \text{Cases } CsQ \end{aligned}$$

- Theorem *bisimSubstParPres*:

$$\Psi \triangleright P \sim_s Q \implies \Psi \triangleright P \parallel R \sim_s Q \parallel R$$

- Theorem *bisimSubstResPres*:

$$\llbracket \Psi \triangleright P \sim_s Q; x \# \Psi \rrbracket \implies \Psi \triangleright (\nu x)P \sim_s (\nu x)Q$$

- Theorem *bisimSubstBangPres*:

$$\llbracket \Psi \triangleright P \sim_s Q; \text{guarded } P; \text{guarded } Q \rrbracket \implies \Psi \triangleright !P \sim_s !Q$$

2.3.4 Structural equivalence – Theorem 11

- Theorem *bisimCasePushRes*:

$$x \# \text{map fst } Cs \implies \Psi \triangleright (\nu x)(\text{Cases } Cs) \sim \text{Cases map } (\lambda(\varphi, P). (\varphi, (\nu x)P)) Cs$$

- Theorem *bisimInputPushRes*:

$$\llbracket x \# M; x \# \text{vec}; x \# N \rrbracket \implies \Psi \triangleright (\nu x)(M(\lambda * \text{vec } N).P) \sim M(\lambda * \text{vec } N).(\nu x)P$$

- Theorem *bisimOutputPushRes*:

$$\llbracket x \# M; x \# N \rrbracket \implies \Psi \triangleright (\nu x)(M\langle N \rangle.P) \sim M\langle N \rangle.(\nu x)P$$

- Theorem *bisimParAssoc*:

$$\Psi \triangleright P \parallel Q \parallel R \sim P \parallel (Q \parallel R)$$

- Theorem *bisimParComm*:

$$\Psi \triangleright P \parallel Q \sim Q \parallel P$$

- Theorem *bisimResNil*:

$$\Psi \triangleright (\nu x)\mathbf{0} \sim \mathbf{0}$$

- Theorem *bisimScopeExt*:

$$x \# P \implies \Psi \triangleright (\nu x)(P \parallel Q) \sim P \parallel (\nu x)Q$$

- Theorem *bisimResComm*:

$$\Psi \triangleright (\nu x)((\nu y)P) \sim (\nu y)((\nu x)P)$$

- Theorem *bangExt*:

$$\textit{guarded } P \implies \Psi \triangleright !P \sim P \parallel !P$$

- Theorem *bisimParNil*:

$$\Psi \triangleright P \parallel \mathbf{0} \sim P$$

2.3.5 Support of processes in broadcast transitions – Lemma 14

- Theorem *brInputTermSupp*:

$$\Psi \triangleright P \mapsto \imath K(N) \prec P' \implies \textit{supp } K \subseteq \textit{supp } P$$

- Theorem *brOutputTermSupp*:

$$\Psi \triangleright P \mapsto R\textit{BrOut } K ((\nu *xvec)N \prec' P') \implies \textit{supp } K \subseteq \textit{supp } P$$

end

end

References

- [1] Jesper Bengtson. Psi-calculi in Isabelle. *Arch. Formal Proofs*, 2012. https://www.isa-afp.org/entries/Psi_Calculi.html, Formal proof development.
- [2] Jesper Bengtson, Magnus Johansson, Joachim Parrow, and Björn Victor. Psi-calculi: a framework for mobile processes with nominal data and logic. *Log. Methods Comput. Sci.*, 7(1), 2011.
- [3] Jesper Bengtson, Joachim Parrow, and Tjark Weber. Psi-calculi in Isabelle. *J. Autom. Reason.*, 56(1):1–47, 2016.

- [4] Johannes Borgström, Shuqin Huang, Magnus Johansson, Palle Raabjerg, Björn Victor, Johannes Åman Pohjola, and Joachim Parrow. Broadcast Psi-calculi with an application to wireless protocols. In Gilles Barthe, Alberto Pardo, and Gerardo Schneider, editors, *Software Engineering and Formal Methods - 9th International Conference, SEFM 2011, Montevideo, Uruguay, November 14-18, 2011. Proceedings*, volume 7041 of *Lecture Notes in Computer Science*, pages 74–89. Springer, 2011.
- [5] Johannes Borgström, Shuqin Huang, Magnus Johansson, Palle Raabjerg, Björn Victor, Johannes Åman Pohjola, and Joachim Parrow. Broadcast psi-calculi with an application to wireless protocols. *Softw. Syst. Model.*, 14(1):201–216, 2015.