Chapter 1

Examples

1.1 GC1 is a garbage-collector

We prove by induction on a proof of $A \rightarrow_{GC1} B$ that $A \rightarrow_{GC1} B$ implies $(A, B) \in GC$.

Rule 1 We have $A = \circ(x).P$ and $B = \emptyset$. Hence, we have $A[\emptyset]$. By definition, $(A, \emptyset) \in GC$.

Rule 2 See rule 1.

Rule 3 Let us consider $R$, the smallest relation such that, for each $P$ such that $P \leftrightarrow a$, if $(\nu a : N)(P[a(x)Q] \rightarrow^* R$ and $R \equiv (\nu a_1 : N_1, \ldots, a : N, a_n : N_n)(P'[a(x)Q]$ then $(R, (\nu a_1 : N_1, \ldots, a : N, a_n : N_n)P') \in \mathcal{R}$.

Let us suppose that $R$ is a barbed simulation. Note first that if $(\nu a : N)(P[a(x)Q] \rightarrow^* R$ and $R \equiv (\nu a_1 : N_1, \ldots, a : N, a_n : N_n)(P'[a(x)Q]$, we have $P' \leftrightarrow a$.

Let us consider $C, D$ such that $CD$. By definition of $R$, for some $P'$, we have $C \equiv C'$ where $C' = (\nu a_1 : N_1, \ldots, a : N, a_n : N_n)(P'[a(x)Q]$ and $D$ must be $(\nu a_1 : N_1, \ldots, a : N, a_n : N_n)P'$. Let us now suppose that $C \downarrow \zeta$. As bars are defined up-to structural congruence, we also have $C' \downarrow \zeta$. Then $\zeta \notin \{a_1, \ldots, a_n\}$ as these names are hidden by $\nu$ and we have either $P' \downarrow \zeta$ or $a(x) \downarrow \zeta$. As $a(x) \downarrow \zeta$ implies $\zeta = a$, which is absurd, we deduce that $P' \downarrow \zeta$. Since $\zeta \notin \{a_1, \ldots, a_n\}$, we also have $D \downarrow \zeta$.

Let us now consider $E$ such that $C \rightarrow E$. As $P' \leftrightarrow a$ the transition from $C$ to $E$ cannot imply any communication on name $a$. Hence, we have $E \equiv (\nu b_1 : N_1, \ldots, a : N, b_m : N_m)(P''[a(x)Q]$ with $E'' \leftrightarrow a$. Hence, by definition, $(E, (\nu b_1 : N_1, \ldots, a : N, b_m : N_m)P'') \in \mathcal{R}$. From the same transition, we also have $D \leftrightarrow (\nu \ldots, a : N, b_m : N_m)P''$.

Hence $R$ is a barbed simulation. Consequently, for $a, b, x P$ and $Q$ such that rule (3), we have $((\nu a : N)(P[a(x)Q], (\nu a : N)P) \in \mathcal{R}$, hence $(\nu a : N)(P[a(x)Q] \ll (\nu a : N)P$. By definition of $GC$, we deduce that $((\nu a : N)(P[a(x)Q], (\nu a : N)P)$ is in GC. Which proves that only garbage-collectable terms are actually garbage-collected.

Rule 4 See Rule 3.

Rule 5 Let us suppose that $(\nu a : T)(P[Q] \rightarrow_{GC1} (\nu a : T)P$. By induction hypothesis, we have $((\nu a : T)(P[Q], (\nu a : T)P) \in GC$. Hence, by definition, we also have $((\nu a : T)(P[Q], (\nu a : T)P) \in GC$. Which proves the case.

We conclude by induction.
1.2 Typing the bounded memory manager

Let us write

\[
\begin{align*}
N_t &= [K_t, z_t], e_t
\end{align*}
\]

\[
\begin{align*}
N_c &= K_c, e_c, e_c
\end{align*}
\]

\[
\begin{align*}
N_r &= [N_r, z_r], e_r
\end{align*}
\]

\[
\begin{align*}
\Gamma(alloc) &= [N_r, z_a], e_a
\end{align*}
\]

Typing \(\Box\)

\[
\Gamma, l : N_t, r : N_r, c : N_c; \emptyset \vdash \Box : u_1 \quad \text{By T-Nil}
\]

\[
\Rightarrow \Gamma, l : N_t, r : N_r, c : N_c; \emptyset \vdash \Box : u_1 - z_t \quad \text{By T-Write}
\]

where \(u_1 - z_t \geq 0\)

Typing \(\Box\)\(\bowtie\)

\[
\Gamma, l : N_t, r : N_r, c : N_c; \emptyset \vdash \Box : \tau(c) : u_1 - z_t - e_t \quad \text{By T-Finalize2}
\]

where \(u_1 - z_t - e_t \geq 0\)

Typing \(\tau(c)\)

\[
\Gamma, l : N_t, r : N_r, c : N_c; \emptyset \vdash \tau(c) : u_2 - z_r \quad \text{By T-Write}
\]

where \(u_2 - z_r \geq 0\)

Typing \(\Box\)\(\bowtie\)\(\tau(c)\)

\[
\Gamma, l : N_t, r : N_r, c : N_c; c \vdash \Box \vdash \Box : u_1 - z_t - e_t \quad \text{See above}
\]

\[
\Gamma, l : N_t, r : N_r, c : N_c; \emptyset \vdash \tau(c) : u_2 - z_r \quad \text{See above}
\]

\[
\Rightarrow \Gamma, l : N_t, r : N_r, c : N_c; c \vdash \Box \vdash \Box : u_1 + u_2 - z_t - e_t - z_r \quad \text{By T-PAR}
\]

Typing \((\nu c : N_c)\)\(\ldots\)

\[
\Gamma, l : N_t, r : N_r, c : N_c; c \vdash (\nu c : N_c)((\Box)\bowtie \tau(c)) : u_1 + u_2 - z_t - z_r \quad \text{See above}
\]

\[
\Rightarrow \Gamma, l : N_t, r : N_r; \emptyset \vdash (\nu c : N_c)((\Box)\bowtie \tau(c)) : u_1 + u_2 - z_t - z_r \quad \text{By T-New}
\]

Typing \(l()\)\(\nu c : N_c)\)\(\ldots\)

\[
\Gamma, l : N_t, r : N_r; \emptyset \vdash (\nu c : N_c)((\Box)\bowtie \tau(c)) : u_1 + u_2 - z_t - z_r \quad \text{See above}
\]

\[
\Rightarrow \Gamma, l : N_t, r : N_r; \emptyset \vdash l()\nu c : N_c)\ldots : u_1 + u_2 - z_t - z_r \quad \text{By T-Read}
\]

Typing alloc\(r()\)\(\nu c : N_c)\)\(\ldots\)

\[
\Rightarrow \Gamma, l : N_t; \emptyset \vdash alloc(r())\nu c : N_c)\ldots : u_1 + u_2 - z_r + z_a \quad \text{By T-Read}
\]

Typing !alloc\(r()\)\(\nu c : N_c)\)\(\ldots\)

\[
\begin{align*}
\text{provided} & \quad u_1 + u_2 - z_r + z_a = 0
\end{align*}
\]

\[
\Rightarrow \Gamma, l : N_t; \Lambda \vdash !alloc(r())\nu c : N_c)\ldots : 1 \quad \text{By T-Repl}
\]

Typing \(\nu l : N_l)\)\(\ldots\)\(\Box)\ldots\)

\[
\Gamma, l : N_t; \Lambda \vdash !alloc(r())\nu c : N_c)\ldots : 0
\]

\[
\Gamma, l : N_l; \Lambda \vdash \Box : u_3 - z_l
\]

\[
\Rightarrow \Gamma; \Lambda \vdash (\nu l : N_l)\ldots : e_1 + n \cdot (u_3 - z_l) \quad \text{By T-New}
\]

**First typing** The first typing specifies that

\[
\begin{align*}
N_t &= [\nu l, -1], 1
\end{align*}
\]

\[
\begin{align*}
N_c &= -1
\end{align*}
\]

\[
\begin{align*}
\Gamma(alloc) &= [[N_c, 0], 0, 0], -
\end{align*}
\]
Hence $z_l = -1$, $e_l = 1$, $e_c = 1$, $z_a = 0$, $z_r = 0$. We then have $\Gamma; \Lambda \vdash (\nu l : N_l)! \cdots : 1 + n \cdot (u_1 + 1)$ with the following conditions:

\[
\begin{aligned}
&u_1 + 1 \geq 0 \\
u_1 \geq 0 \\
u_2 - z_r \geq 0 \\
u_1 + \nu_2 - z_r = 0 \\
u_3 - z_l \geq 0
\end{aligned}
\]

With $u_1 = u_2 = u_3 = 0$, we obtain $\Gamma; \Lambda \vdash (\nu l : N_l)! \cdots : n + 1$.

**Second typing** The second typing specifies that

\[
\begin{aligned}
N_l &= [+0], 1 \\
N_c &= [0], 1 \\
\Gamma(\text{alloc}) &= [[N_c, 0], 0, -1], .
\end{aligned}
\]

Hence $z_l = 0$, $e_l = 1$, $e_c = 1$, $z_a = -1$, $z_r = 0$. We then have $\Gamma; \Lambda \vdash (\nu l : N_l)! \cdots : 1 + n \cdot u_1$ with the following conditions:

\[
\begin{aligned}
&u_1 \geq 0 \\
u_1 - 1 \geq 0 \\
u_2 \geq 0 \\
u_1 + \nu_2 - 1 = 0 \\
u_3 - z_l \geq 0
\end{aligned}
\]

With $u_1 = 1$ and $u_2 = u_3 = 0$, we obtain $\Gamma; \Lambda \vdash (\nu l : N_l)! \cdots : 1$. 

3
Chapter 2

Properties

2.1 Subject Reduction

\[
\begin{align*}
\text{T-Nil} & : \quad \Gamma; \Lambda \vdash 0 : t \\
\text{T-New} & : \quad \gamma : (K, e); \Lambda \vdash P : t_P \quad x \notin \Lambda \\
\text{T-Par} & : \quad \Gamma; \Lambda \vdash P : t_P \quad \Gamma; \Lambda \vdash Q : t_Q \quad \Gamma; \Lambda \vdash P \cup Q : t_P + t_Q \\
\text{T-Repl} & : \quad \Gamma; \emptyset \vdash P : 0 \\
\text{T-Sum} & : \quad \Gamma; \Lambda \vdash P : t_P \\
\text{T-Read} & : \quad \Gamma; \Lambda \vdash c(y) : Q : t_Q \\
\text{T-Write} & : \quad \Gamma; \Lambda \vdash c(y) : Q : t_Q + z \\
\end{align*}
\]

2.1.1 Lemmas

**Lemma 1 (Substitution)** If we have \( \Gamma, x : T; \Lambda \vdash P : U \) and \( \Gamma; \Lambda \vdash a : T \), then, if \( x \notin \Lambda \), then \( \Gamma; \Lambda \vdash \text{Subs}(P, x, a) : U \).

Proof by induction on a structure of a proof of \( \Gamma, x : T; \Lambda \vdash P : U \).

**T-Nil** Base case, trivial.

**T-Par** By inheritance.

**T-New** \((\nu y)P\) if \( x = y \), base case, trivial.

**T-New** \((\nu y)P\) if \( x \neq y \), by induction hypothesis.

**T-Finalize1** \((\forall y)P\) if \( x \neq y \), directly by induction hypothesis.

**T-Finalize1** \((\forall y)P\) if \( x = y \), we use the induction hypothesis, the type of \( y \) is unchanged. Since \( x \notin \Lambda \), no difficulty.

**T-Finalize2** \((\forall y)P\) if \( x \neq y \), directly by induction hypothesis.

**T-Finalize2** \((\forall y)P\) if \( x = y \), we use the induction hypothesis, the type of \( y \) is unchanged, no difficulty.

**T-Read** \( c(y)P \) if \( x \neq y \) and \( x \neq c \), directly by induction hypothesis.
T-Read  \(c(y)P\) : if \(x = c\), we use the induction hypothesis, the type of \(c\) is unchanged, no difficulty.

T-Read  \(c(y)P\) : \(x = y\) implies a double binding, which is in contradiction with the hypothesis.

T-Write By induction hypothesis, no difficulty.

T-Repl Directly by induction hypothesis.

T-Sum Directly by induction hypothesis.

Lemma 2 (Resource expansion) If \(\Gamma; \Lambda \vdash P : \text{Proc}(t)\) and \(u \geq t\) then \(\Gamma; \Lambda \vdash P : \text{Proc}(u)\).

By induction on the structure of a proof of \(\Gamma; \Lambda \vdash P : \text{Proc}(u)\).

T-Nil Base case, trivial.

T-ReadW Base case, trivial.

T-WriteW Base case, trivial.

T-Repl Base case, trivial.

T-Par Directly, by induction hypothesis.

T-New By induction hypothesis, without difficulty.

T-Finalize1 By induction hypothesis, without difficulty.

T-Finalize2 By induction hypothesis, without difficulty.

T-Read By induction hypothesis, without difficulty.

T-Write By induction hypothesis, without difficulty.

T-Sum By induction hypothesis, without difficulty.

Lemma 3 (Weakening) If \(\Gamma; \Lambda \vdash P : U\) and \(n \notin \text{fn}(P)\), then \(\Gamma, n : A; \Lambda \vdash P : U\).

By induction on the structure of a proof of \(\Gamma, n : A; \Lambda \vdash P : U\). Since only the names which appear in a process are used to type it, this is trivial.

Lemma 4 (Strengthening) If \(\Gamma, n : A; \Lambda \vdash P : U\) and \(n \notin \text{fn}(P)\), then \(\Gamma; \Lambda \vdash P : U\).

By induction on the structure of a proof of \(\Gamma; \Lambda \vdash P : U\). Since only the names which appear in a process are used to type it, this is trivial.

Lemma 5 (Strategies weakening) If \(\Gamma; \Lambda \vdash P : U\) and \(n \notin \text{bv}(P)\) then \(\Gamma; \Lambda, n \vdash P\).

By induction on the structure of a proof of \(\Gamma; \Lambda \vdash P : U\).

T-Nil Base case: since \(\text{bv}(P) = \emptyset\), there is nothing to prove.

T-New when \(P = (\nu x)Q\) with \(x = n\). Base case: since \(n \in \text{bv}(P)\), there is nothing to prove.

T-Finalize1 when \(P = (\lambda x)Q\) with \(x = n\). Base case: since \(n \in \Lambda\), there is nothing to prove.

T-Read when \(P = c(x)Q\) with \(x = n\). Base case: since \(n \in \text{bv}(P)\), there is nothing to prove.

T-Repl Base case, as \(\Lambda\) is not constrained.

T-Par Since \(\text{bv}(Q|R) = \text{bv}(Q) \cup \text{bv}(R)\), this is a direct consequence of the induction hypothesis.

T-New when \(P = (\nu x)Q\) with \(x \neq n\), this is a direct consequence of the induction hypothesis.

T-Finalize1 when \(P = (\lambda x)Q\) with \(x \neq n\), this is a direct consequence of the induction hypothesis.
T-Finalize2 when \( P = (\exists x)Q \) with \( x \neq n \), this is a direct consequence of the induction hypothesis.

T-Finalize2 when \( P = (\exists x)Q \) with \( x = n \). We then have \( \Gamma; \Lambda \vdash Q : t_Q \) and \( x \notin \Lambda \). Let us write \( \Gamma(x) = \omega, e \). By resource expansion, we also have \( \Gamma; \Lambda \vdash Q : t_Q + e \). Since \( t_Q + e \geq e \) and \( x \notin \Lambda \), we may use T-Finalize1 and conclude \( \Gamma; \Lambda, x \vdash (\exists x)Q : U \).

T-Read when \( P = c(x)Q \) with \( x \neq n \), this is a direct consequence of the induction hypothesis.

T-Write this is a direct consequence of the induction hypothesis.

T-WriteW this is a direct consequence of the induction hypothesis.

T-Sum this is a direct consequence of the induction hypothesis.

Lemma 6 (Strategies strengthening) If \( \Gamma; \Lambda, n \vdash P : U \) and \( n \notin \text{fv}(P) \) then \( \Gamma; \Lambda \vdash P : U \).

By induction on a proof of \( \Gamma; \Lambda \vdash P \).

T-Nil Base case, trivial.

T-Par Let us write \( P = Q|R \). Since \( \text{fv}(Q|R) = \text{fv}(Q) \cup \text{fv}(R) \), we have \( n \notin \text{fv}(P) \) and \( n \notin \text{fv}(Q) \). Thus, it is sufficient to apply the induction hypothesis to \( Q \) and \( R \).

T-New When \( P = (\nu x : T)Q \) with \( x \neq n \). Since \( \text{fv}((\nu x : T)Q) = \text{fv}(Q) \setminus \{x\} \), we have \( n \notin \text{fv}(Q) \). Thus, it is sufficient to apply the induction hypothesis to \( Q \).

T-New When \( P = (\nu x : T)Q \) with \( x = n \). Using T-New, we know that \( x \notin \text{fv}(P) \). Thus there is nothing to prove.

T-Finalize1 When \( P = (\exists x)Q \) with \( x \neq n \). Since \( n \notin \text{fv}(P) \), we also have \( n \notin \text{fv}(Q) \). Thus, it is sufficient to apply the induction hypothesis to \( Q \).

T-Finalize1 When \( P = (\exists x)Q \) with \( x = n \). We then have \( n \in \Lambda \), which proves the case.

T-Finalize2 When \( P = (\exists x)Q \) with \( x \neq n \). We then have \( n \notin \text{fv}(Q) \). Thus, it is sufficient to apply the induction hypothesis to \( Q \).

T-Finalize2 When \( P = (\exists x)Q \) with \( x = n \). We then have \( n \in \text{fv}(P) \). Thus, there is nothing to prove.

T-Read When \( P = c(x)Q \) with \( x \neq n \). We then have \( n \notin \text{fv}(Q) \). Thus, it is sufficient to apply the induction hypothesis to \( Q \).

T-Read When \( P = c(x)Q \) with \( x = n \). This cannot happen as it would imply \( n \notin \Lambda, n \).

T-Write When \( P = \overline{c}(x)Q \). As \( \text{fv}(P) = \text{fv}(Q) \cup \{c, x\} \), we also have \( n \notin \text{fv}(Q) \). Thus, it is sufficient to apply the induction hypothesis to \( Q \).

T-Repl Base case. Trivial.

T-Sum As T-PAR

2.1.2 Type of congruent terms

Lemma 7 (Type of congruent terms) If \( \Gamma; \Lambda \vdash A : T \) and \( A \equiv B \) then \( \Gamma; \Lambda \vdash B : T \).
S-EQUIV-REFL \[ P \equiv P \]
S-EQUIV-SYM \[ Q \equiv P \] \[ P \equiv Q \]
S-EQUIV-TRANS \[ P \equiv Q \] \[ Q \equiv R \] \[ P \equiv R \]
S-PAR-ASSOC \[ P|(Q|R) \equiv (P|Q)|R \]
S-PAR-NIL \[ P\{0\} \equiv P \]
S-SUM-COMM \[ P + Q \equiv Q + P \]
S-SUM-NIL \[ P + 0 \equiv P \]
S-PAR-ASSOC \[ P \mid (Q \mid R) \equiv (P \mid Q) \mid R \]
S-PAR-NIL \[ P \mid \emptyset \equiv P \]
S-SUM-COMM \[ P \mid Q \equiv Q \mid P \]
S-SUM-NIL \[ P \mid \emptyset \equiv P \]
S-STRUCT-PAR \[ P \equiv Q \]
S-NEW-REN \[ (\nu x : N)A \equiv (\nu y : N).\text{Subst}(P, x, y) \]
S-RCV-REN \[ c(x).P \equiv c(y).\text{Subst}(P, x, y) \]
S-NEW-COMM \[ (\nu x : N)(\nu y : U)A \equiv (\nu y : U)(\nu x : N)A \]
S-NEW-PAR \[ (\nu x : N)(P|Q) \equiv P(\nu x : N)Q \]
S-FIN-PAR \[ (\forall x : T)\{\forall x : T\} \equiv (\forall x : T)(P|Q) \]

By induction hypothesis on the structure of a proof of \( A \equiv B \) or \( B \equiv A \).

**S-EQUIV-REFL**

Base case. This is trivial.

**S-EQUIV-TRANS**

By induction hypothesis. This is trivial.

**S-PAR-ASSOC**

Let us write \( A = P|(Q|R) \) and \( B = (P|Q)|R \). We will then prove that if \( \Gamma; \Lambda \vdash A : T \) then \( \Gamma; \Lambda \vdash B : T \) and that if \( \Gamma; \Lambda \vdash B : T \) then \( \Gamma; \Lambda \vdash A : T \).

\[
\begin{align*}
\Gamma; \Lambda_P \vdash P & : t_P \\
\Gamma; \Lambda_Q \vdash Q & : t_Q \\
\Gamma; \Lambda_R \vdash R & : t_R
\end{align*}
\]

**Typing \( A \)**

**Typing \( Q|R \)**

\[
\begin{align*}
\Gamma; \Lambda_Q \vdash Q & : t_Q & \text{By hypothesis}
\end{align*}
\]

\[
\begin{align*}
\Gamma; \Lambda_R \vdash R & : t_R & \text{By hypothesis}
\end{align*}
\]

\[
\Rightarrow \Gamma; \Lambda_Q \cup \Lambda_R \vdash Q|R & : t_Q + t_R & \text{By T-Par if } \Lambda_Q \cap \Lambda_R = \emptyset
\]

**Typing \( A \)**

\[
\begin{align*}
\Gamma; \Lambda_P \vdash P & : t_P & \text{By hypothesis}
\end{align*}
\]

\[
\begin{align*}
\Gamma; \Lambda_Q \cup \Lambda_R \vdash Q|R & : t_Q + t_R & \text{See above if } \Lambda_P \cap (\Lambda_Q \cup \Lambda_R) = \emptyset
\end{align*}
\]

\[
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \cup \Lambda_R \vdash P|(Q|R) & : t_P + t_Q + t_R & \text{By T-Par}
\]

7
Typing $B$

$$
\begin{array}{ll}
\text{Typing } P|Q \\
\Gamma; \Lambda_P \vdash P : \quad t_P & \text{By hypothesis} \\
\Gamma; \Lambda_Q \vdash Q : \quad t_Q & \text{By hypothesis} \\
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \vdash P|Q : \quad t_P + t_Q & \text{By T-Par}
\end{array}
$$

Typing $B$

$$
\begin{array}{ll}
\Gamma; \Lambda_R \vdash R : \quad t_R & \text{By hypothesis} \\
\Gamma; \Lambda_P \cup \Lambda_Q \vdash P|Q : \quad t_P + t_Q & \text{See above} \\
\text{if } \quad \Lambda_R \cap (\Lambda_P \cup \Lambda_Q) = \emptyset \\
\Rightarrow \Gamma; \Lambda_R \cup \Lambda_P \cup \Lambda_Q \vdash (P|Q)|R : \quad t_P + t_Q + t_R & \text{By T-Par}
\end{array}
$$

We then have if $\Gamma; \Lambda \vdash A : T$ then $\Gamma; \Lambda \vdash B : T$. The other way is identical.

S-PAR-COMM

This is purely symmetrical. Usual proof.

S-PAR-NIL

Let us write $A = P|0$ and $B = P$.

Whenever $P$ may be typed in $\Gamma; \Lambda$, we will write

$$
\Gamma; \Lambda_P \vdash P : \quad t_P
$$

Let us suppose that $A$ may be typed with type $t_A$ in $\Gamma; \Lambda$. By T-Nil and T-Par, necessarily, we have $\Lambda = \Lambda_P \cup \Lambda_0$ for some $\Lambda_0$ and $P$ typeable in $\Gamma; \Lambda$. We then have the following typing:

$$
\Rightarrow \Gamma; \Lambda_0 \vdash 0 : \quad t_0 & \text{By T-Nil}
$$

$$
\Rightarrow \Gamma; \Lambda_0 \vdash 0 : \quad t_0 & \text{See above}
$$

$$
\text{if } \quad \Lambda_0 \cup \Lambda_0 = \emptyset \\
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_0 \vdash P|0 : \quad t_P + t_0 & \text{By T-Par}
$$

Moreover, we have $t_A \geq t_P$. Hence, using the Useless Strategies lemma, we have $\Gamma; \Lambda_P \cup \Lambda_0 \vdash P : t_P$.

By Expansion lemma, as $t_A \geq t_P$, we also have $\Gamma; \Lambda_P \cup \Lambda_0 \vdash P : t_A$.

Conversely, let us now suppose $P$ typeable with type $t_P$ in $\Gamma; \Lambda$. From the same typing, with $\Lambda_0 = \emptyset$ and $t_0 = 0$, we conclude that $A$ is typeable with type $t_P$.

The case is proved.

S-BANG

Let us write $A = !P$ and $B = !P|P$.

Let us write $\Gamma; \Lambda_P \vdash P : t_P$. 

}\"
Typing $A$

<table>
<thead>
<tr>
<th>Typing $!P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda P \vdash P : t_P$</td>
</tr>
</tbody>
</table>

if $\Lambda_P = 0$

$\Rightarrow \Gamma; \Lambda \vdash !P : t \quad \text{T-Repl}$

Typing $B$

<table>
<thead>
<tr>
<th>Typing $!P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda P \vdash P : t_P$</td>
</tr>
</tbody>
</table>

if $\Lambda_P = 0$

$\Rightarrow \Gamma; \Lambda \vdash !P : t \quad \text{T-Repl}$

<table>
<thead>
<tr>
<th>Typing $B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda \vdash !P : t \quad \text{See above}$</td>
</tr>
<tr>
<td>$\Gamma; \emptyset \vdash P : 0$</td>
</tr>
</tbody>
</table>

$\Rightarrow \vdash !P|P : t \quad \text{T-Par}$

Let us suppose $A$ typeable in $\Gamma; \Lambda$ with type $t$. From T-Repl, we then have $\Lambda_P = \emptyset$ and $t_P = 0$. From the typing of $B$, we conclude that $B$ is typeable in $\Gamma; \Lambda$ and may have any type, including $t$.

Let us now suppose $B$ typeable in $\Gamma; \Lambda$ with type $t$. Similarly, we have $\Lambda_P = \emptyset$ and $t_P = 0$. From the typing of $A$, we see that $A$ is typeable in $\Gamma; \Lambda$ and may have any type, including $t$.

The case is proved.

**S-SUM-COMM**

Trivial.

**S-STRUCT-PAR**

Let us write $A = P|R$ and $B = Q|R$ with $P \equiv Q$. As $A$ is typeable and the only possible way of typing $A$ is R-Par, we have $\Gamma; \Lambda_P \vdash P : t_P$ and $\Gamma; \Lambda_R \vdash P : t_R$ with $t_P + t_R = t$. By induction hypothesis, as $P \equiv Q$, we also have $\Gamma; \Lambda_P \vdash Q : t_P$. By R-Par, we conclude $\Gamma; \Lambda_P \cup \Lambda_R \vdash Q|R : t_R + t_P$. The case is proved.

Going from $B$ to $A$ is similar.

**S-NEW-COMM**

Trivial.

**S-NEW-PAR**

Let us write $A = (\nu x : T)(P|Q)$ and $B = P|(\nu x : T)Q$ where $T = \omega, e$.

From $A$ to $B$ – typing $A$ Whenever $\Gamma; \Lambda \vdash A : t$, let us write $\Gamma = \Gamma_{PQ}, x : T$, $\Gamma_{PQ}; \Lambda_P \vdash P : t_P$, $\Gamma_{PQ}; \Lambda_Q \vdash Q : t_Q$
Typing $P|Q$

$\Gamma_{PQ}; \Lambda_P \vdash P : t_P$ \hspace{1cm} By hypothesis

$\Gamma_{PQ}; \Lambda_Q \vdash Q : t_Q$ \hspace{1cm} By hypothesis

$\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \vdash P|Q : t_P + t_Q$ \hspace{1cm} By T-Par

where $\Lambda_P \cap \Lambda_Q = \emptyset$

Typing $A$

$\Gamma_{PQ}; \Lambda_P \cup \Lambda_Q \vdash P|Q : t_P + t_Q$ \hspace{1cm} See above

$\Rightarrow \Gamma; (\Lambda_P \cup \Lambda_Q) \setminus \{x\} \vdash A : t_P + t_Q + e$ \hspace{1cm} By T-New

where $x \in \Lambda_P \cup \Lambda_Q$

From $A$ to $B$ – typing $B$  
As $x \in \Lambda_P \cup \Lambda_Q$ and $\Lambda_P \cap \Lambda_Q = \emptyset$, we deduce that either $x \in \Lambda_P \setminus \Lambda_Q$ or $x \in \Lambda_Q \setminus \Lambda_P$.

If we have $x \in \Lambda_P \setminus \Lambda_Q$, since $x \notin fv(P)$, by “Strategies strengthening”, we deduce that $\Gamma_{PQ}; \Lambda_P \setminus \{x\} \vdash P : t_P$. By Strategies weakening, we may also deduce $\Gamma_{PQ}; \Lambda_Q, x \vdash Q : t_Q$. We may then suppose without loss of generality that $x \in \Lambda_Q \setminus \Lambda_P$.

Typing $(\nu x : T)Q$

$\Gamma_{PQ}; \Lambda_Q \vdash Q : t_Q$ \hspace{1cm} By hypothesis

$\Rightarrow \Gamma; \Lambda_Q \setminus \{x\} \vdash (\nu x : T)Q : t_Q + e$ \hspace{1cm} By T-New

Typing $P$

$\Gamma_{PQ}; \Lambda_P \vdash P : t_P$ \hspace{1cm} By hypothesis

$\Rightarrow \Gamma; \Lambda_P \vdash P : t_P$ \hspace{1cm} Using lemma Weakening

Typing $B$

$\Gamma; \Lambda_P \vdash P : t_P$ \hspace{1cm} See above

$\Rightarrow \Gamma; (\Lambda_P \cup \Lambda_Q) \setminus \{x\} \vdash P : t_P + t_Q + e$ \hspace{1cm} By T-Par

Hence proving the case.

From $B$ to $A$ – typing $B$  
Whenever $\Gamma; \Lambda \vdash A : t$, let us write $\Gamma; \Lambda_P \vdash P : t_P, \Gamma, x : T; \Lambda_Q \vdash Q : t_Q$. 

10
Typing \((\nu x : T)Q\)
\[
\Gamma, x : T; \Lambda_Q \vdash Q : t_Q \quad \text{By hypothesis}
\]
\[
\Rightarrow \Gamma; \Lambda_Q \setminus \{x\} \vdash (\nu x : T)Q : t_Q \quad \text{By T-New}
\]
where \(x \in \Lambda_Q\)

Typing \(B\)
\[
\Gamma; \Lambda_P \vdash P : t_P \quad \text{By hypothesis}
\]
\[
\Gamma; \Lambda_Q \setminus \{x\} \vdash (\nu x : T)Q : t_Q \quad \text{See above}
\]
\[
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \setminus \{x\} \vdash B : t_P + t_Q \quad \text{By T-Par}
\]
where \(\Lambda_P \cap (\Lambda_Q \setminus \{x\}) = \emptyset\)

From \(B\) to \(A - \text{typing} A\) As previously, it is sufficient to prove the case where \(x \notin \Lambda_P\). We will suppose \(\Lambda_P \cap \Lambda_Q = \emptyset\)

Typing \(P\)
\[
\Gamma; \Lambda_P \vdash P : t_P \quad \text{By hypothesis}
\]
\[
\Rightarrow \Gamma; x : T; \Lambda_P \vdash P : t_P \quad \text{Using lemma Weakening}
\]

Typing \(P|Q\)
\[
\Gamma; x : T; \Lambda_P \vdash P : t_P \quad \text{See above}
\]
\[
\Gamma, x : T; \Lambda_Q \vdash Q : t_Q \quad \text{By hypothesis}
\]
\[
\Rightarrow \Gamma; x : T; \Lambda_P \cup \Lambda_Q \vdash P|Q : t_P + t_Q \quad \text{By T-Par}
\]
since \(\Lambda_P \cap \Lambda_Q = \emptyset\)

Typing \((\nu x : T)(P|Q)\)
\[
\Gamma; x : T; \Lambda_P \cup \Lambda_Q \vdash P|Q : t_P + t_Q \quad \text{See above}
\]
since \(x \in \Lambda_P \cup \Lambda_Q\)
\[
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \setminus \{x\} \vdash (\nu x : T)(P|Q) : t_P + t_Q + e \quad \text{By T-New}
\]

The case is proved.

**S-Equiv-Refl**

Let us suppose that our proof of \(A \equiv B\) ends with \(\text{S-eqv-refl}\). We then have a shorter proof of \(B \equiv A\). By induction hypothesis, since \(\Gamma; \Lambda \vdash A : T\) and \(B \equiv A\), we also have \(\Gamma; \Lambda \vdash B : T\). By invoking \(\text{S-eqv-refl}\), we conclude.

2.1.3 Subject Reduction

**Theorem 1 (Subject Reduction)** \(\text{Si } \Gamma; \Lambda \vdash A : T \text{ et } A \rightarrow B \text{ alors } \Gamma; \Lambda \vdash B : T\).

By induction on the structure of a proof of \(A \rightarrow B\).

**R-Comm**

Let us write \(A = \pi(b)P|a(x)Q\) with \(a \neq \circ\) and \(B = P|\mathrm{Subs}(Q,x,b)\).

We will use the following notations:
\[
\begin{align*}
\Gamma; \Lambda_P & \vdash P : t_P \\
\Gamma, x : (C_b, e_b); \Lambda_Q & \vdash Q : t_Q \\
\Gamma(a) & = [C_b, e_b, z_a, e_a] \\
\Gamma(b) & = C_b, e_b
\end{align*}
\]
Typing A

\[
\begin{array}{c|c|c}
\text{Typing } \pi(b)P \\
\hline
\Gamma; \Lambda_P & \vdash P : t_P & \text{By hypothesis} \\
\hline
\Gamma(a) = [C_b, e_b, z_a], e_a & \text{By hypothesis} \\
\Gamma(b) = C_b, e_b & \text{By hypothesis} \\
\Rightarrow \Gamma; \Lambda_P & \vdash \pi(b)P : t_Q - z & \text{By T-Write} \\
& \text{where } t_Q - z \geq 0 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{Typing } a(x)Q \\
\hline
\Gamma, x : C_b & \vdash Q : t_Q & \text{By hypothesis} \\
\hline
\Gamma(a) = [C_b, e_b, z_a], e_a & \text{By hypothesis} \\
\Rightarrow \Gamma; \Lambda_Q & \vdash a(x)Q : t_Q + z & \text{By T-Read} \\
& \text{where } x \notin \Lambda_Q \\
& \quad t_Q + z \geq 0 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{Typing } A \\
\hline
\Gamma; \Lambda_P & \vdash \pi(b)P : t_P + z & \text{See above} \\
\Gamma; \Lambda_Q & \vdash a(x)Q : t_Q - z & \text{See above} \\
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q & \vdash A : t_P + t_Q & \text{By T-PAR} \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{Typing } A \\
\hline
\Gamma; \Lambda_P & \vdash \pi(b)P : t_P + z & \text{See above} \\
\Gamma; \Lambda_Q & \vdash a(x)Q : t_Q - z & \text{See above} \\
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q & \vdash A : t_P + t_Q & \text{By T-PAR} \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{Typing } B \\
\hline
\Gamma, x : C_b; \Lambda_Q & \vdash Q : t_Q & \text{By hypothesis} \\
\hline
\Gamma(b) = C_b, e_b & \text{By hypothesis} \\
\Rightarrow \Gamma; \Lambda_Q & \vdash \text{Subs}(Q, x, b) : t_Q & \text{Using lemma Substitution} \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{Typing } B \\
\hline
\Gamma; \Lambda_P & \vdash P : t_P & \text{By hypothesis} \\
\hline
\Gamma; \Lambda_P & \vdash \text{Subs}(Q, x, b) : t_Q & \text{See above} \\
\text{since } \Lambda_P \cap \Lambda_Q = \emptyset & \text{See above} \\
\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q & \vdash B : t_P + t_Q & \text{By T-PAR} \\
\end{array}
\]

The case is proved.

\textbf{R-COMM-SUM}

The case is almost identical.

\textbf{R-PAR}

Let us write \( A = P|R \) and \( B = Q|R \) where \( P \rightarrow Q \). Let us also write

12
\[ \Gamma; \Lambda_P \vdash P : t_P \]
\[ \Gamma; \Lambda_R \vdash R : t_R \]

**Typing A**

<table>
<thead>
<tr>
<th>Typing (P</th>
<th>R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\Gamma; \Lambda_P)</td>
<td>(\vdash P : t_P) By hypothesis</td>
</tr>
<tr>
<td>(\Gamma; \Lambda_R)</td>
<td>(\vdash R : t_R) By hypothesis</td>
</tr>
<tr>
<td>(\Rightarrow \Gamma; \Lambda_P \cup \Lambda_R)</td>
<td>(\vdash P</td>
</tr>
</tbody>
</table>

where \(\Lambda_P \cap \Lambda_R = \emptyset\)

**Typing B**

| Typing \(Q\) |  
|----------------|---|
| \(\Gamma; \Lambda_P\) | \(\vdash P : t_P\) By hypothesis |

since \(P \rightarrow Q\)

<table>
<thead>
<tr>
<th>Typing (Q</th>
<th>R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\Gamma; \Lambda_Q)</td>
<td>(\vdash Q : t_Q) See above</td>
</tr>
<tr>
<td>(\Gamma; \Lambda_R)</td>
<td>(\vdash R : t_R) By hypothesis</td>
</tr>
</tbody>
</table>

since \(\Lambda_P \cap \Lambda_R = \emptyset\)

| \(\Rightarrow \Gamma; \Lambda_P \cup \Lambda_R\) | \(\vdash Q|R : t_P + t_R\) By T-PAR |

The case is proved.

**R-NEW**

Directly by induction hypothesis.

**R-EQUIV**

Directly from the lemma.

**R-AUTOCLEAN**

Strengthening + Resource expansion + Useless strategies.

**R-FINALIZE**

Let us write \(A = (\nu x : T)(\forall x)P\) and \(B = \text{Subs}(P, x, \diamond)\).  

\[ \Gamma, x : T; \Lambda_P \vdash P : t_P \]
Typing $A$

\[
\begin{array}{c|c|c}
\hline
\text{Typing } (\forall x)P \\
\hline
\Gamma, x : T; \Lambda_P \vdash P : t_P & \text{By hypothesis} \\
\hline
(\Gamma, x : T)(x) = T & \text{By hypothesis} \\
\hline
\Gamma, x : T; \Lambda_P, x \vdash (\forall x)P : t_P - e & \text{By T-Finalize1} \\
\hline
\end{array}
\]

where \( t_P \geq e \)
\[ x \notin \Lambda_P \]

Typing $A$

\[
\begin{array}{c|c|c}
\hline
\text{Typing } (\forall x)P \\
\hline
\Gamma, x : T; \Lambda_P, x \vdash (\forall x)P : t_P - e & \text{See above} \\
\hline
\Gamma; \Lambda_P \vdash (\forall x : T)(\forall x)P : t_P & \text{By T-New} \\
\hline
\end{array}
\]

where \[ x \notin \Lambda_P \]

Typing $B$

\[
\begin{array}{c|c|c}
\hline
\text{Typing } B \\
\hline
\Gamma, x : T; \Lambda_P \vdash P : t_P & \text{By hypothesis} \\
\hline
\Gamma(\diamond) = T & \text{By hypothesis} \\
\hline
\Gamma(\diamond) = T & \text{By hypothesis} \\
\hline
\Rightarrow \Gamma; \Lambda_P \vdash Subs(P, x, \diamond) : t_P & \text{Using lemma Substitution} \\
\hline
\end{array}
\]

The case is proved.

**R-FIN-PAR**

Let us write $A = (\forall x)P|(\forall x)Q$ and $B = (\forall x)(P|Q)$

\[
\begin{align*}
\Gamma; \Lambda_P & \vdash P : t_P \\
\Gamma; \Lambda_Q & \vdash Q : t_Q \\
\Gamma(x) : K_x, c_x
\end{align*}
\]
Typing $A$ – first strategy

<table>
<thead>
<tr>
<th>Typing ($\forall x)P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_P \vdash$ $P : t_P$</td>
</tr>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_P, x \vdash (\forall x)P : t_P - e$</td>
</tr>
<tr>
<td>where $t_P \geq e$ $x \notin \Lambda_P$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typing ($\forall x)Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_Q \vdash$ $Q : t_Q$</td>
</tr>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_Q \vdash (\forall x)Q : t_Q$</td>
</tr>
<tr>
<td>where $x \notin \Lambda_Q$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typing $A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_P, x \vdash (\forall x)P : t_P - e$</td>
</tr>
<tr>
<td>$\Gamma; \Lambda_Q \vdash (\forall x)Q : t_Q$</td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q, x \vdash (\forall x)P(\forall x)Q : t_P + t_Q - e$</td>
</tr>
<tr>
<td>where $(\Lambda_P, x) \cap \Lambda_Q = \emptyset$</td>
</tr>
</tbody>
</table>

Typing $A$ – second strategy

<table>
<thead>
<tr>
<th>Typing ($\forall x)Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_Q \vdash$ $Q : t_Q$</td>
</tr>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_Q, x \vdash (\forall x)Q : t_Q - e$</td>
</tr>
<tr>
<td>where $t_Q \geq e$ $x \notin \Lambda_Q$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typing ($\forall x)P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_P \vdash$ $P : t_P$</td>
</tr>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_P \vdash (\forall x)P : t_P$</td>
</tr>
<tr>
<td>where $x \notin \Lambda_P$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typing $A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_Q, x \vdash (\forall x)Q : t_Q - e$</td>
</tr>
<tr>
<td>$\Gamma; \Lambda_P \vdash (\forall x)P : t_P$</td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q, x \vdash (\forall x)Q(\forall x)Q : t_P + t_Q - e$</td>
</tr>
<tr>
<td>where $(\Lambda_Q, x) \cap \Lambda_P = \emptyset$</td>
</tr>
</tbody>
</table>
Typing $A$ – third strategy

<table>
<thead>
<tr>
<th>Typing ($\forall x)P$</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_P \vdash P : t_P$</td>
<td>By hypothesis</td>
<td></td>
</tr>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
<td>By hypothesis</td>
<td></td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_P \vdash (\forall x)P : t_P$</td>
<td>By T-Finalize2</td>
<td></td>
</tr>
<tr>
<td>where $x \notin \Lambda_P$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typing ($\forall x)Q$</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_Q \vdash Q : t_Q$</td>
<td>By hypothesis</td>
<td></td>
</tr>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
<td>By hypothesis</td>
<td></td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_Q \vdash (\forall x)Q : t_Q$</td>
<td>By T-Finalize2</td>
<td></td>
</tr>
<tr>
<td>where $x \notin \Lambda_Q$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typing $A$</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma; \Lambda_P \vdash (\forall x)P : t_P$</td>
<td>See above</td>
<td></td>
</tr>
<tr>
<td>$\Gamma; \Lambda_Q \vdash (\forall x)Q : t_Q$</td>
<td>See above</td>
<td></td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \vdash (\forall x)P</td>
<td>(\forall x)Q : t_P + t_Q$</td>
<td>By T-Par</td>
</tr>
<tr>
<td>where $\Lambda_P \cap \Lambda_Q = \emptyset$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Typing $B$ to match the two first strategies

| Typing $P|Q$ |            |            |
|----------------|------------|------------|
| $\Gamma; \Lambda_P \vdash P : t_P$ | By hypothesis |
| $\Gamma; \Lambda_Q \vdash Q : t_Q$ | By hypothesis |
| $\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \vdash P|Q : t_P + t_Q$ | By T-Par |
| since $\Lambda_P \cap \Lambda_Q = \emptyset$ |

<table>
<thead>
<tr>
<th>Typing $B$</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma(x) : K_x, e_x$</td>
<td>By hypothesis</td>
<td></td>
</tr>
<tr>
<td>since $t_P + t_Q \geq e$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \notin \Lambda_P \cup \Lambda_Q$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Rightarrow \Gamma; (\Lambda_P \cup \Lambda_Q), x \vdash (\forall x)(P</td>
<td>Q) : t_P + t_Q - e$</td>
<td>By T-Finalize1</td>
</tr>
</tbody>
</table>
Typing $B$ to match the third strategy

|                | Typing $P|Q$ |
|----------------|-----------|
| $\Gamma; \Lambda_P \vdash$ | $P : t_P$ | By hypothesis |
| $\Gamma; \Lambda_Q \vdash$ | $Q : t_Q$ | By hypothesis |

$\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \vdash \ P|Q : t_P + t_Q$  

By $T$-$\text{PAR}$

<table>
<thead>
<tr>
<th></th>
<th>Typing $B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Gamma(x) :$</td>
<td>$K_x, e_x$</td>
</tr>
</tbody>
</table>

$\Rightarrow \Gamma; \Lambda_P \cup \Lambda_Q \vdash (\forall x)(P|Q) : t_P + t_Q$  

By $T$-$\text{FINALIZE2}$

The case is proved.