

The Calculus of Linear Constructions — Technical Report

Qiancheng Fu

December 29, 2021

Contents

1	Introduction	2
2	Syntax of CLC (<code>clc_ast.v</code>)	2
3	Reduction and Equality of CLC (<code>clc_ast.v</code>)	2
4	Confluence of CLC (<code>clc_confluence.v</code>)	2
4.1	Parallel Reduction	2
4.2	Reduction Lemmas	3
4.3	Equality Lemmas	4
4.4	Parallel Reduction Lemmas	4
4.5	Confluence Theorem	5
4.6	Corollaries of Confluence	5
5	Context of CLC (<code>clc_context.v</code>)	6
5.1	Merge Lemmas	6
5.2	Restriction and Purity Lemmas	7
6	Subtyping of CLC (<code>clc_subtype.v</code>)	8
6.1	Subtyping Lemmas	8
7	Typing of CLC (<code>clc_typing.v</code>)	9
8	Inversion Lemmas of CLC (<code>clc_inversion.v</code>)	10
9	Weakening Lemmas of CLC (<code>clc_weakening.v</code>)	11
9.1	Properties of <i>agreeR</i>	11
9.2	Weakening Theorem	11
10	Substitution Lemmas of CLC (<code>clc_substitution.v</code>)	12
10.1	Properties of <i>agreeS</i>	12
10.2	Substitution Lemma	12
10.3	Corollaries of Substitution	12
11	Typing Validity of CLC (<code>clc_validity.v</code>)	13
12	Subject Reduction of CLC (<code>clc_soundness.v</code>)	13

1 Introduction

This extended report is meant to accompany our paper of the same title. Here, we describe the meta-theory of CLC and CILC and their proofs in detail. All the results presented here have been formalized and proven correct in the Coq Proof Assistant.

2 Syntax of CLC (`clc_ast.v`)

i $::= 0 \mid 1 \mid 2 \dots$ universe levels

s, t $::= U \mid L$ sorts

m, n, A, B, M $::= U_i \mid L_i \mid x$ expressions
 $\quad \mid (x :_s A) \rightarrow B$
 $\quad \mid (x :_s A) \multimap B$
 $\quad \mid \lambda x :_s A. n$
 $\quad \mid m \ n$

3 Reduction and Equality of CLC (`clc_ast.v`)

$$\begin{array}{c}
\frac{m_1 \rightsquigarrow^* n \quad m_2 \rightsquigarrow^* n}{m_1 \equiv m_2 : A} \text{JOIN} \quad \frac{}{(\lambda x :_s A. m) \ n \rightsquigarrow m[n/x]} \text{STEP-}\beta \quad \frac{A \rightsquigarrow A'}{\lambda x :_s A. m \rightsquigarrow \lambda x :_s A'. m} \text{STEP-}\lambda L \\
\\
\frac{m \rightsquigarrow m'}{\lambda x :_s A. m \rightsquigarrow \lambda x :_s A. m'} \text{STEP-}\lambda R \quad \frac{A \rightsquigarrow_p A'}{(x :_s A) \rightarrow B \rightsquigarrow (x :_s A') \rightarrow B} \text{STEP-L}\rightarrow \\
\\
\frac{B \rightsquigarrow_p B'}{(x :_s A) \rightarrow B \rightsquigarrow (x :_s A) \rightarrow B'} \text{STEP-R}\rightarrow \quad \frac{A \rightsquigarrow_p A'}{(x :_s A) \multimap B \rightsquigarrow (x :_s A') \multimap B} \text{STEP-L}\multimap \\
\\
\frac{B \rightsquigarrow_p B'}{(x :_s A) \multimap B \rightsquigarrow (x :_s A) \multimap B'} \text{STEP-R}\multimap \quad \frac{m \rightsquigarrow m'}{m \ n \rightsquigarrow m' \ n} \text{STEP-APPL} \quad \frac{n \rightsquigarrow n'}{m \ n \rightsquigarrow m \ n'} \text{STEP-APPR}
\end{array}$$

4 Confluence of CLC (`clc_confluence.v`)

4.1 Parallel Reduction

To prove the confluence property of CLC, we employ the standard technique utilizing parallel reductions.

$$\begin{array}{c}
\frac{}{x \rightsquigarrow_p x} \text{PSTEP-VAR} \qquad \frac{}{s_i \rightsquigarrow_p s_i} \text{PSTEP-SORT} \qquad \frac{A \rightsquigarrow_p A' \quad m \rightsquigarrow_p m'}{\lambda x :_s A.m \rightsquigarrow_p \lambda x :_s A'.m'} \text{PSTEP-}\lambda \\
\\
\frac{m \rightsquigarrow_p m' \quad n \rightsquigarrow_p n'}{m \ n \rightsquigarrow_p m' \ n'} \text{PSTEP-APP} \qquad \frac{m \rightsquigarrow_p m' \quad n \rightsquigarrow_p n'}{(\lambda x :_s A.m) \ n \rightsquigarrow_p m'[n'/x]} \text{PSTEP-}\beta \\
\\
\frac{A \rightsquigarrow_p A' \quad B \rightsquigarrow_p B'}{(x :_s A) \rightarrow B \rightsquigarrow_p (x :_s A') \rightarrow B'} \text{PSTEP-}\rightarrow \qquad \frac{A \rightsquigarrow_p A' \quad B \rightsquigarrow_p B'}{(x :_s A) \multimap B \rightsquigarrow_p (x :_s A') \multimap B'} \text{PSTEP-}\multimap
\end{array}$$

4.2 Reduction Lemmas

Here, we prove some simple lemmas concerning \rightsquigarrow , \rightsquigarrow^* and substitution.

Definition 4.1. For a term m and a map σ from variables to terms, let $m[\sigma]$ be the term obtained by applying σ uniformly to all free variables in m .

Definition 4.2. For maps σ, τ from variables to terms, we say that σ reduces to τ if for any variable x there exists a reduction $(\sigma \ x) \rightsquigarrow^* (\tau \ x)$. We write $\sigma \rightsquigarrow^* \tau$ when it is clear from context that σ, τ are maps and not terms.

Lemma 4.1. For terms m, n and a map σ from variables to terms, if there exist a step $m \rightsquigarrow n$, then there exists a step $m[\sigma] \rightsquigarrow n[\sigma]$.

Proof. By induction on the derivation of $m \rightsquigarrow n$. □

Lemma 4.2. For terms m_1, m_2, n_1, n_2 , if there exists reductions $m_1 \rightsquigarrow^* m_2$ and $n_1 \rightsquigarrow^* n_2$, then there exists reduction $(m_1 \ n_1) \rightsquigarrow^* (m_2 \ n_2)$.

Proof. By transitivity of \rightsquigarrow^* and applying rules STEP-APPL, STEP-APPR. □

Lemma 4.3. For terms A_1, A_2, m_1, m_2 and sort s , if there exists reductions $A_1 \rightsquigarrow^* A_2$ and $m_1 \rightsquigarrow^* m_2$, then there exists reduction $\lambda x :_s A_1.m_1 \rightsquigarrow^* \lambda x :_s A_2.m_2$.

Proof. By transitivity of \rightsquigarrow^* and applying rules STEP-AL, STEP-AR. □

Lemma 4.4. For terms A_1, A_2, B_1, B_2 and sort s , if there exists reductions $A_1 \rightsquigarrow^* A_2$ and $B_1 \rightsquigarrow^* B_2$, then there exists reduction $(x :_s A_1) \rightarrow B_1 \rightsquigarrow^* (x :_s A_2) \rightarrow B_2$.

Proof. By transitivity of \rightsquigarrow^* and applying rules STEP-L \rightarrow , STEP-R \rightarrow . □

Lemma 4.5. For terms A_1, A_2, B_1, B_2 and sort s , if there exists reductions $A_1 \rightsquigarrow^* A_2$ and $B_1 \rightsquigarrow^* B_2$, then there exists reduction $(x :_s A_1) \multimap B_1 \rightsquigarrow^* (x :_s A_2) \multimap B_2$.

Proof. By transitivity of \rightsquigarrow^* and applying rules STEP-L \multimap , STEP-R \multimap . □

Lemma 4.6. For terms m, n and a map σ from variables to terms, if there exist a reduction $m \rightsquigarrow^* n$, then there exists a reduction $m[\sigma] \rightsquigarrow^* n[\sigma]$.

Proof. By induction on the derivation of \rightsquigarrow^* , the transitivity of \rightsquigarrow^* and Lemma 4.1. □

Lemma 4.7. For maps σ, τ from variables to terms, if there is a map reduction $\sigma \rightsquigarrow^* \tau$, then for any term m there is a reduction $m[\sigma] \rightsquigarrow^* m[\tau]$.

Proof. By induction on the structure of m , applying Lemmas 4.2, 4.3, 4.4, 4.5. □

4.3 Equality Lemmas

Here, we prove some simple lemmas concerning \rightsquigarrow^* , \equiv and substitution.

Definition 4.3. For maps σ, τ from variables to terms, we say that σ is equal to τ if for any variable x there exists an equality $(\sigma x) \equiv (\tau x)$. We write $\sigma \equiv \tau$ when it is clear from context that σ, τ are maps and not terms.

Lemma 4.8. For any map f from terms to terms, if for any terms m, n such that $m \rightsquigarrow n$ implies $f m \equiv f n$, then for any terms m, n equality $m \equiv n$ implies $f m \equiv f n$.

Proof. By the properties of the transitive reflexive closure \rightsquigarrow^* and that \equiv is an equivalence relation. \square

Lemma 4.9. For terms m_1, m_2, n_1, n_2 , if there exists equalities $m_1 \equiv m_2$ and $n_1 \equiv n_2$, then there exists equality $(m_1 n_1) \equiv (m_2 n_2)$.

Proof. By transitivity of \equiv and applying rules JOIN, STEP-APPL, STEP-APPR. \square

Lemma 4.10. For terms A_1, A_2, m_1, m_2 and sort s , if there exists equalities $A_1 \equiv A_2$ and $m_1 \equiv m_2$, then there exists equality $\lambda x :_s A_1.m_1 \equiv \lambda x :_s A_2.m_2$.

Proof. By transitivity of \equiv and applying rules JOIN, STEP- λ L, STEP- λ R. \square

Lemma 4.11. For terms A_1, A_2, B_1, B_2 and sort s , if there exists equalities $A_1 \equiv A_2$ and $B_1 \equiv B_2$, then there exists equality $(x :_s A_1) \rightarrow B_1 \equiv (x :_s A_2) \rightarrow B_2$.

Proof. By transitivity of \equiv and applying rules JOIN, STEP-L \rightarrow , STEP-R \rightarrow . \square

Lemma 4.12. For terms A_1, A_2, B_1, B_2 and sort s , if there exists equalities $A_1 \equiv A_2$ and $B_1 \equiv B_2$, then there exists equality $(x :_s A_1) \multimap B_1 \equiv (x :_s A_2) \multimap B_2$.

Proof. By transitivity of \equiv and applying rules JOIN, STEP-L \multimap , STEP-R \multimap . \square

Lemma 4.13. For terms m, n and map σ from variables to terms, if there is equality $m \equiv n$, then there is equality $m[\sigma] \equiv n[\sigma]$.

Proof. By Lemmas 4.8 and 4.1. \square

Lemma 4.14. For maps σ, τ from variables to terms and term m , if there is map equality $\sigma \equiv \tau$, then there is equality $m[\sigma] \equiv m[\tau]$.

Proof. By induction on the structure of m , applying Lemmas 4.9, 4.10, 4.11, 4.12. \square

Lemma 4.15. For terms m_1, m_2, n , if there is equality $m_1 \equiv m_2$, then there is equality $n[m_1/x] \equiv n[m_2/x]$ for any variable $x \in FV(n)$.

Proof. This is a special case of Lemma 4.14 where σ maps x to m_1 and τ maps x to m_2 . \square

4.4 Parallel Reduction Lemmas

Definition 4.4. For maps σ, τ from variables to terms, we say σ parallel reduces to τ if for any variable x there exists a parallel reduction $(\sigma x) \rightsquigarrow_p (\tau x)$. We write $\sigma \rightsquigarrow_p \tau$ when it is clear from context that σ, τ are maps and not terms.

Lemma 4.16. For any term m , there exists a reflexive parallel reduction $m \rightsquigarrow_p m$.

Proof. By induction on the structure of m . \square

Lemma 4.17. For any map σ from variables to terms, there exists a reflexive parallel map reduction $\sigma \rightsquigarrow_p \sigma$.

Proof. By Definition 4.4 and Lemma 4.16. \square

Lemma 4.18. For any terms m, n , if there exists step $m \rightsquigarrow n$, then there exists a parallel reduction $m \rightsquigarrow_p n$.

Proof. By induction on the derivation of $m \rightsquigarrow n$ and Lemma 4.16. \square

Lemma 4.19. *For terms m, n , if there exists parallel reduction $m \rightsquigarrow_p n$, then there exists a reduction $m \rightsquigarrow^* n$.*

Proof. By induction on the derivation of $m \rightsquigarrow_p n$, utilizing the transitive property of \rightsquigarrow^* and Lemmas 4.2, 4.3, 4.4, 4.5, 4.6, 4.7. \square

Lemma 4.20. *For terms m, n and map σ from variables to terms, if there exists parallel reduction $m \rightsquigarrow_p n$, there exists parallel reduction $m[\sigma] \rightsquigarrow_p n[\sigma]$.*

Proof. By induction on the derivation of $m \rightsquigarrow_p n$ and Lemma 4.16. \square

Lemma 4.21. *For terms m, n and maps σ, τ from variables to terms, if there exists parallel reduction $m \rightsquigarrow_p n$ and parallel map reduction $\sigma \rightsquigarrow_p \tau$, there exists parallel reduction $m[\sigma] \rightsquigarrow_p n[\tau]$.*

Proof. By induction on the derivation of $m \rightsquigarrow_p n$. \square

Lemma 4.22. *For terms m_1, m_2, n , if there is parallel reduction $m_1 \rightsquigarrow_p m_2$, then there is parallel reduction $n[m_1/x] \rightsquigarrow_p n[m_2/x]$ for any variable $x \in FV(n)$.*

Proof. By Lemma 4.16, this is a special case of Lemma 4.21 where σ maps x to m_1 and τ maps x to m_2 . \square

4.5 Confluence Theorem

We first show that \rightsquigarrow_p satisfies the diamond property. Using the diamond property, we ultimately prove the confluence theorem.

Lemma 4.23. *CLC term reduction has the diamond property. For terms m, m_1, m_2 , if there are parallel reductions $m \rightsquigarrow_p m_1$ and $m \rightsquigarrow_p m_2$, then there exists term m' such that $m_1 \rightsquigarrow_p m'$ and $m_2 \rightsquigarrow_p m'$.*

Proof. By induction on the derivation of $m \rightsquigarrow_p m_1$. Each case in the induction specializes m appearing in $m \rightsquigarrow_p m_2$, allowing one to invert its derivation in a syntax directed way and apply the induction hypothesis. The difficult cases are due to PSTEP- β as it concerns substitution, so Lemma 4.21 is used to push these cases through. \square

Lemma 4.24. *Strip lemma. For terms m, m_1, m_2 , if there is parallel reduction $m \rightsquigarrow_p m_1$ and reduction $m \rightsquigarrow^* m_2$, then there exists term m' such that $m_1 \rightsquigarrow^* m'$ and $m_2 \rightsquigarrow_p m'$.*

Proof. By induction on the derivation of $m \rightsquigarrow_p m_1$, utilizing transitivity of \rightsquigarrow^* and Lemmas 4.18, 4.19, 4.23. \square

Theorem 4.25. *CLC term reduction is confluent. For terms m, m_1, m_2 , if there are reductions $m \rightsquigarrow^* m_1$ and $m \rightsquigarrow^* m_2$, then there exists term m' such that $m_1 \rightsquigarrow^* m'$ and $m_2 \rightsquigarrow^* m'$.*

Proof. By induction on the derivation of $m \rightsquigarrow^* m_1$, utilizing transitivity of \rightsquigarrow^* and Lemmas 4.18, 4.19, 4.24. \square

4.6 Corollaries of Confluence

The following results are all corollaries of confluence, proven using a combination of induction, transitivity and confluence. These corollaries allow us to refute false reductions and equalities in future proofs.

Corollary 4.25.1. *For a universe s_i and term m , if there is reduction $s_i \rightsquigarrow^* m$, then $m = s_i$.*

Corollary 4.25.2. *For variable x and term m , if there is reduction $x \rightsquigarrow^* m$, then $m = x$.*

Corollary 4.25.3. *For terms A, B, m and sort s , if there is reduction $(x :_s A) \rightarrow B \rightsquigarrow^* m$, then there exists A', B' such that there are reductions $A \rightsquigarrow^* A'$, $B \rightsquigarrow^* B'$ and $m = (x :_s A') \rightarrow B'$.*

Corollary 4.25.4. For terms A, B, m and sort s , if there is reduction $(x :_s A) \multimap B \rightsquigarrow^* m$, then there exists A', B' such that there are reductions $A \rightsquigarrow^* A'$, $B \rightsquigarrow^* B'$ and $m = (x :_s A') \multimap B'$.

Corollary 4.25.5. For terms A, m, n and sort s , if there is reduction $\lambda x :_s A. m \rightsquigarrow^* n$, then there exists A', m' such that there are reductions $A \rightsquigarrow^* A'$, $m \rightsquigarrow^* m'$ and $n = \lambda x :_s A'. m'$.

Corollary 4.25.6. For sorts s, t and levels i, j , if there is equality $s_i \equiv t_j$, then there is $s = t$ and $i = j$.

Corollary 4.25.7. For terms A_1, A_2, B_1, B_2 and sorts s, t , if there is equality $(x :_s A_1) \rightarrow B_1 \equiv (x :_t A_2) \rightarrow B_2$, then there are equalities $A_1 \equiv A_2$, $B_1 \equiv B_2$ and $s = t$.

Corollary 4.25.8. For terms A_1, A_2, B_1, B_2 and sorts s, t , if there is equality $(x :_s A_1) \multimap B_1 \equiv (x :_t A_2) \multimap B_2$, then there are equalities $A_1 \equiv A_2$, $B_1 \equiv B_2$ and $s = t$.

5 Context of CLC (`clc_context.v`)

Contexts of CLC are of the form $x_1 :_{s_1} A_1, x_2 :_{s_2} A_2, \dots, x_k :_{s_k} A_k$ where each free variable x_i is assigned a type A_i and sort s_i . Contexts will be referred to by meta variables Γ and Δ .

$$\begin{array}{c}
\frac{}{\epsilon \vdash} \text{WF-}\epsilon \qquad \frac{\Gamma \vdash \quad \bar{\Gamma} \vdash A : U_i}{\Gamma, x :_U A \vdash} \text{WF-U} \qquad \frac{\Gamma \vdash \quad \bar{\Gamma} \vdash A : L_i}{\Gamma, x :_L A \vdash} \text{WF-L} \\
\\
\frac{}{|\epsilon|} \text{PURE-}\epsilon \qquad \frac{|\Gamma| \quad \Gamma \vdash A : U_i}{|\Gamma, x :_U A|} \text{PURE-U} \\
\\
\frac{}{\epsilon \ddag \epsilon \ddag \epsilon} \text{MERGE-}\epsilon \qquad \frac{\Gamma_1 \ddag \Gamma_2 \ddag \Gamma}{\Gamma_1, x :_U A \ddag \Gamma_2, x :_U A \ddag \Gamma, x :_U A} \text{MERGE-U} \\
\\
\frac{\Gamma_1 \ddag \Gamma_2 \ddag \Gamma \quad x \notin \Gamma_2}{\Gamma_1, x :_L A \ddag \Gamma_2 \ddag \Gamma, x :_L A} \text{MERGE-L1} \qquad \frac{\Gamma_1 \ddag \Gamma_2 \ddag \Gamma \quad x \notin \Gamma_1}{\Gamma_1 \ddag \Gamma_2, x :_L A \ddag \Gamma, x :_L A} \text{MERGE-L2}
\end{array}$$

5.1 Merge Lemmas

Since weakening and contraction rules will not be allowed on restricted variables, it is necessary to have lemmas that enable the manipulation of contexts.

Lemma 5.1. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \ddag \Gamma_2 \ddag \Gamma$, then there is $\Gamma_2 \ddag \Gamma_1 \ddag \Gamma$.

Proof. By induction on the derivation of $\Gamma_1 \ddag \Gamma_2 \ddag \Gamma$. □

Lemma 5.2. For any context Γ , if there is $|\Gamma|$, then there is $\Gamma \ddag \Gamma \ddag \Gamma$.

Proof. By induction on the derivation of $|\Gamma|$. □

Lemma 5.3. For any context Γ , there is $\bar{\Gamma} \ddag \Gamma \ddag \Gamma$.

Proof. By induction on the structure of Γ . □

Lemma 5.4. For any context Γ , there is $\Gamma \ddag \bar{\Gamma} \ddag \Gamma$.

Proof. By induction on the structure of Γ . □

Lemma 5.5. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \ddag \Gamma_2 \ddag \Gamma$ and $|\Gamma|$, then there is $|\Gamma_1|$ and $|\Gamma_2|$.

Proof. By induction on the derivation of $\Gamma_1 \ddag \Gamma_2 \ddag \Gamma$. □

Lemma 5.6. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$ and $|\Gamma_1|$, then there is $\Gamma = \Gamma_2$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

Lemma 5.7. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$ and $|\Gamma_2|$, then there is $\Gamma = \Gamma_1$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

Lemma 5.8. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$, and also $|\Gamma_1|$, $|\Gamma_2|$, then there is $|\Gamma|$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

Lemma 5.9. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$, and also $|\Gamma_1|$, $|\Gamma_2|$, then there is $\Gamma_1 = \Gamma_2$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

Lemma 5.10. For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$, then there is $\overline{\Gamma_1} = \overline{\Gamma}$ and $\overline{\Gamma_2} = \overline{\Gamma}$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

Lemma 5.11. For any context Γ , there is $\overline{\Gamma} \dagger \overline{\Gamma} \dagger \overline{\Gamma}$.

Proof. By induction on the structure of Γ . □

5.2 Restriction and Purity Lemmas

Lemma 5.12. For any context Γ , there is $\overline{\Gamma} = \overline{\overline{\Gamma}}$.

Proof. By induction on the structure of Γ . □

Lemma 5.13. For any context Γ , if there is $|\Gamma|$, then there is $\Gamma = \overline{\Gamma}$.

Proof. By induction on the structure of Γ . □

Lemma 5.14. For any context Γ , there is $|\overline{\Gamma}|$.

Proof. By induction on the structure of Γ . □

Lemma 5.15. For any context Γ , variable x and type A , if there is $x :_U A \in \Gamma$, then there is $x :_U A \in \overline{\Gamma}$.

Proof. By induction on the derivation of $x :_U A \in \Gamma$. □

Lemma 5.16. For any context Γ , variable x and type A , there is $x :_L A \notin \overline{\Gamma}$.

Proof. By induction on the structure of Γ . □

Lemma 5.17. For contexts $\Gamma_1, \Gamma_2, \Gamma, \Delta_1, \Delta_2$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$ and $\Delta_1 \dagger \Delta_2 \dagger \Gamma_1$, then there exists Δ such that $\Delta_1 \dagger \Gamma_2 \dagger \Delta$ and $\Delta \dagger \Delta_2 \dagger \Gamma$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

Lemma 5.18. For contexts $\Gamma_1, \Gamma_2, \Gamma, \Delta_1, \Delta_2$, if there is $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$ and $\Delta_1 \dagger \Delta_2 \dagger \Gamma_1$, then there exists Δ such that $\Delta_2 \dagger \Gamma_2 \dagger \Delta$ and $\Delta_1 \dagger \Delta \dagger \Gamma$.

Proof. By induction on the derivation of $\Gamma_1 \dagger \Gamma_2 \dagger \Gamma$. □

6 Subtyping of CLC (clc_subtype.v)

The cumulativity relation (\preceq) is the smallest binary relation over terms such that

1. \preceq is a partial order with respect to equality.

- (a) If $A \equiv B$, then $A \preceq B$.
- (b) If $A \preceq B$ and $B \preceq A$, then $A \equiv B$.
- (c) If $A \preceq B$ and $B \preceq C$, then $A \preceq C$.

2. $U_0 \preceq U_1 \preceq U_2 \preceq \dots$

3. $L_0 \preceq L_1 \preceq L_2 \preceq \dots$

4. If $A_1 \equiv A_2$ and $B_1 \preceq B_2$,
then $(x :_s A_1) \rightarrow B_1 \preceq (x :_s A_2) \rightarrow B_2$

5. If $A_1 \equiv A_2$ and $B_1 \preceq B_2$,
then $(x :_s A_1) \multimap B_1 \preceq (x :_s A_2) \multimap B_2$

Here, we give an inductive definition of the cumulativity relation (\preceq) that is suitable for writing proofs.

$$\begin{array}{c}
 \frac{}{A \prec A} \prec\text{-REFL} \qquad \frac{i_1 \leq i_2}{s_{i_1} \prec s_{i_2}} \prec\text{-SORT} \qquad \frac{B_1 \prec B_2}{(x :_s A) \rightarrow B_1 \prec (x :_s A) \rightarrow B_2} \prec\rightarrow \\
 \\
 \frac{B_1 \prec B_2}{(x :_s A) \multimap B_1 \prec (x :_s A) \multimap B_2} \prec\multimap \qquad \frac{A' \prec B' \quad A \equiv A' \quad B \equiv B'}{A \preceq B} \prec\preceq
 \end{array}$$

6.1 Subtyping Lemmas

Lemma 6.1. *For terms A, B , if there is $A \prec B$, then there is $A \preceq B$.*

Proof. By $\prec\preceq$ and the reflexivity of equality \equiv . □

Lemma 6.2. *For terms A, B, C , if there is $A \prec B$ and $B \equiv C$, then there is $A \preceq C$.*

Proof. By $\prec\preceq$ and the transitivity of equality \equiv . □

Lemma 6.3. *For terms A, B, C , if there is $A \equiv B$ and $B \prec C$, then there is $A \preceq C$.*

Proof. By $\prec\preceq$ and the transitivity of equality \equiv . □

Lemma 6.4. *For terms A, B , if there is $A \equiv B$, then there is $A \preceq B$.*

Proof. By Lemma 6.3 and $\prec\text{-REFL}$. □

Lemma 6.5. *For term A , there is $A \preceq A$.*

Proof. By Lemma 6.1 and $\prec\text{-REFL}$. □

Lemma 6.6. *For natural numbers i, j and sort s such that $i \leq j$, there is $s_i \preceq s_j$.*

Proof. By Lemma 6.1 and $\prec\text{-SORT}$. □

Lemma 6.7. *For terms A, B, C, D , if there is $A \prec B$, $B \equiv C$ and $C \prec D$, then there is $A \preceq D$.*

Proof. By induction on the derivation of $A \prec B$, definition of \prec and Lemmas 6.1, 6.2, 6.3. □

Lemma 6.8. *For terms A, B, C , if there is $A \preceq B$ and $B \preceq C$, then there is $A \preceq C$.*

Proof. By transitivity of \equiv , rule $\prec\preceq$ and Lemma 6.7. □

Lemma 6.9. For sorts s, t and natural numbers i, j , if there is $s_i \preceq t_j$, then there is $s = t$ and $i \leq j$.

Proof. By transitivity of \equiv and Corollary 4.25.6. \square

Lemma 6.10. For terms A_1, A_2, B_1, B_2 and sorts s, t , if there is $(x :_s A_1) \rightarrow B_1 \preceq (x :_t A_2) \rightarrow B_2$, then there is $A_1 \equiv A_2$ and $B_1 \preceq B_2$ and $s = t$.

Proof. By transitivity of \equiv and Corollary 4.25.7. \square

Lemma 6.11. For terms A_1, A_2, B_1, B_2 and sorts s, t , if there is $(x :_s A_1) \multimap B_1 \preceq (x :_t A_2) \multimap B_2$, then there is $A_1 \equiv A_2$ and $B_1 \preceq B_2$ and $s = t$.

Proof. By transitivity of \equiv and Corollary 4.25.8. \square

Lemma 6.12. For terms A, B and map σ from variables to terms, if there is $A \prec B$, then there is $A[\sigma] \prec B[\sigma]$.

Proof. By induction on the derivation of $A \prec B$ and the definition of \prec . \square

Lemma 6.13. For terms A, B and map σ from variables to terms, if there is $A \preceq B$, then there is $A[\sigma] \preceq B[\sigma]$.

Proof. By rule $\prec\text{-}\preceq$ and Lemmas 4.13, 6.12. \square

7 Typing of CLC (clc_typing.v)

The following rules define well-formed contexts.

$$\frac{}{\epsilon \vdash} \epsilon\text{-OK} \quad \frac{\Gamma \vdash \quad \bar{\Gamma} \vdash A : U_i}{\Gamma, x :_U A \vdash} \text{U-OK} \quad \frac{\Gamma \vdash \quad \bar{\Gamma} \vdash A : L_i}{\Gamma, x :_L A \vdash} \text{L-OK}$$

The typing rules of CLC are presented below.

$$\begin{array}{c} \frac{|\Gamma|}{\Gamma \vdash s_i : U_{i+1}} \text{SORT-AXIOM} \quad \frac{|\Gamma| \quad \Gamma \vdash A : U_i \quad \Gamma, x :_U A \vdash B : s_i}{\Gamma \vdash (x :_U A) \rightarrow B : U_i} \text{U} \rightarrow \\[10pt] \frac{|\Gamma| \quad \Gamma \vdash A : L_i \quad \Gamma \vdash B : s_i \quad x \notin \Gamma}{\Gamma \vdash (x :_L A) \rightarrow B : U_i} \text{L} \rightarrow \quad \frac{|\Gamma| \quad \Gamma \vdash A : U_i \quad \Gamma, x :_U A \vdash B : s_i}{\Gamma \vdash (x :_U A) \multimap B : L_i} \text{U} \multimap \\[10pt] \frac{|\Gamma| \quad \Gamma \vdash A : L_i \quad \Gamma \vdash B : s_i \quad x \notin \Gamma}{\Gamma \vdash (x :_L A) \multimap B : L_i} \text{L} \multimap \quad \frac{|\Gamma_1, \Gamma_2|}{\Gamma_1, x :_U A, \Gamma_2 \vdash x : A} \text{U-VAR} \quad \frac{|\Gamma_1, \Gamma_2|}{\Gamma_1, x :_L A, \Gamma_2 \vdash x : A} \text{L-VAR} \\[10pt] \frac{|\Gamma| \quad \Gamma \vdash (x :_s A) \rightarrow B : t_i \quad \Gamma, x :_s A \vdash n : B}{\Gamma \vdash \lambda x :_s A. n : (x :_s A) \rightarrow B} \lambda \rightarrow \quad \frac{\bar{\Gamma} \vdash (x :_s A) \multimap B : t_i \quad \Gamma, x :_s A \vdash n : B}{\Gamma \vdash \lambda x :_s A. n : (x :_s A) \multimap B} \lambda \multimap \\[10pt] \frac{\Gamma_1 \vdash m : (x :_U A) \rightarrow B \quad |\Gamma_2| \quad \Gamma_2 \vdash n : A \quad \Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma}{\Gamma \vdash m \ n : B[n/x]} \text{APP-U} \rightarrow \\[10pt] \frac{\Gamma_1 \vdash m : (x :_L A) \rightarrow B \quad \Gamma_2 \vdash n : A \quad \Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma}{\Gamma \vdash m \ n : B[n/x]} \text{APP-L} \rightarrow \end{array}$$

$$\begin{array}{c}
\frac{\Gamma_1 \vdash m : (x :_{\mathcal{U}} A) \multimap B \quad |\Gamma_2| \quad \Gamma_2 \vdash n : A \quad \Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma}{\Gamma \vdash m \ n : B[n/x]}_{\text{APP-U}\multimap} \\
\\
\frac{\Gamma_1 \vdash m : (x :_{\mathcal{L}} A) \multimap B \quad \Gamma_2 \vdash n : A \quad \Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma}{\Gamma \vdash m \ n : B[n/x]}_{\text{APP-L}\multimap} \\
\\
\frac{\Gamma \vdash m : A \quad \bar{\Gamma} \vdash B : s_i \quad A \preceq B}{\Gamma \vdash m : B}_{\text{CONVERSION}}
\end{array}$$

8 Inversion Lemmas of CLC (clc_inversion.v)

Lemma 8.1. *For any context Γ and terms A, B, s , if there is $\Gamma \vdash (x :_{\mathcal{U}} A) \rightarrow B : s$, then there exists sort t and natural number i such that $\Gamma \vdash A : U_i$ and $\Gamma, x :_{\mathcal{U}} A \vdash B : t_i$.*

Proof. By induction on the derivation of $\Gamma \vdash (x :_{\mathcal{U}} A) \rightarrow B : s$. □

Lemma 8.2. *For any context Γ and terms A, B, s , if there is $\Gamma \vdash (x :_{\mathcal{L}} A) \rightarrow B : s$, then there exists sort t and natural number i such that $\Gamma \vdash A : L_i$ and $\Gamma \vdash B : t_i$.*

Proof. By induction on the derivation of $\Gamma \vdash (x :_{\mathcal{L}} A) \rightarrow B : s$. □

Lemma 8.3. *For any context Γ and terms A, B, s , if there is $\Gamma \vdash (x :_{\mathcal{U}} A) \multimap B : s$, then there exists sort t and natural number i such that $\Gamma \vdash A : U_i$ and $\Gamma, x :_{\mathcal{U}} A \vdash B : t_i$.*

Proof. By induction on the derivation of $\Gamma \vdash (x :_{\mathcal{U}} A) \multimap B : s$. □

Lemma 8.4. *For any context Γ and terms A, B, s , if there is $\Gamma \vdash (x :_{\mathcal{L}} A) \multimap B : s$, then there exists sort t and natural number i such that $\Gamma \vdash A : L_i$ and $\Gamma \vdash B : t_i$.*

Proof. By induction on the derivation of $\Gamma \vdash (x :_{\mathcal{L}} A) \multimap B : s$. □

Lemma 8.5. *For any context Γ , terms A, n, C and sort s , if there is $\Gamma \vdash \lambda x :_s A.n : C$, then for all terms A', B , sorts s', t and natural number i such that $C \preceq (x :_{s'} A') \rightarrow B$ and $\bar{\Gamma}, x :_{s'} A' \vdash B : t_i$, there is $\Gamma, x :_{s'} A' \vdash n : B$.*

Proof. By induction on the derivation of $\Gamma \vdash \lambda x :_s A.n : C$ and Lemmas 8.1, 8.2. □

Lemma 8.6. *For any context Γ , terms A, n, C and sort s , if there is $\Gamma \vdash \lambda x :_s A.n : C$, then for all terms A', B , sorts s', t and natural number i such that $C \preceq (x :_{s'} A') \multimap B$ and $\bar{\Gamma}, x :_{s'} A' \vdash B : t_i$, there is $\Gamma, x :_{s'} A' \vdash n : B$.*

Proof. By induction on the derivation of $\Gamma \vdash \lambda x :_s A.n : C$ and Lemmas 8.3, 8.4. □

Lemma 8.7. *For any context Γ , terms A, A', B, n , sorts s, s', t and natural number i , if there is $\bar{\Gamma} \vdash (x :_{s'} A') \rightarrow B : t_i$ and $\Gamma \vdash \lambda x :_s A.n : (x :_{s'} A') \rightarrow B$, then there is $\Gamma, x :_{s'} A' \vdash n : B$.*

Proof. Direct consequence of Lemmas 8.1, 8.2 and 8.5. □

Lemma 8.8. *For any context Γ , terms A, A', B, n , sorts s, s', t and natural number i , if there is $\bar{\Gamma} \vdash (x :_{s'} A') \multimap B : t_i$ and $\Gamma \vdash \lambda x :_s A.n : (x :_{s'} A') \multimap B$, then there is $\Gamma, x :_{s'} A' \vdash n : B$.*

Proof. Direct consequence of Lemmas 8.3, 8.4 and 8.6. □

9 Weakening Lemmas of CLC (clc_weakening.v)

Weakening for non-linear types is admissible in CLC. To prove this, we first define an *agreeR* relation between two contexts Γ, Γ' and a mapping ξ from variables to variables.

$$\begin{array}{c} \frac{}{\text{agreeR } \xi \in \epsilon} \text{AGREER-}\epsilon \qquad \frac{\text{agreeR } \xi \Gamma \Gamma' \quad x \notin FV(\Gamma) \cup FV(\Gamma')}{\text{agreeR } (\xi \cup (x, x)) (\Gamma, x :_{\mathcal{U}} A) (\Gamma', x :_{\mathcal{U}} A[\xi])} \text{AGREER-U} \\[10pt] \frac{\text{agreeR } \xi \Gamma \Gamma' \quad x \notin FV(\Gamma) \cup FV(\Gamma')}{\text{agreeR } (\xi \cup (x, x)) (\Gamma, x :_{\mathcal{L}} A) (\Gamma', x :_{\mathcal{L}} A[\xi])} \text{AGREER-L} \qquad \frac{\text{agreeR } \xi \Gamma \Gamma' \quad x \notin FV(\Gamma) \cup FV(\Gamma')}{\text{agreeR } \xi \Gamma (\Gamma', x :_{\mathcal{U}} A)} \text{AGREER-WK} \end{array}$$

9.1 Properties of *agreeR*

Lemma 9.1. *For any context Γ and the identity map id from variables to variables, $\text{agreeR } id \Gamma \Gamma$ is always true.*

Proof. By induction on the structure of Γ and the definition of *agreeR*. □

Lemma 9.2. *For contexts Γ, Γ' and mapping ξ , if there is $\text{agreeR } \xi \Gamma \Gamma'$ and $|\Gamma|$, then there is $|\Gamma'|$.*

Proof. By induction on the derivation of $\text{agreeR } \xi \Gamma \Gamma'$. □

Lemma 9.3. *For contexts Γ, Γ' and mapping ξ , if there is $\text{agreeR } \xi \Gamma \Gamma'$, then there is $\text{agreeR } \xi |\Gamma| |\Gamma'|$.*

Proof. By induction on the derivation of $\text{agreeR } \xi \Gamma \Gamma'$. □

9.2 Weakening Theorem

Lemma 9.4. *For contexts $\Gamma, \Gamma', \Gamma_1, \Gamma_2$ and mapping ξ , if there is $\text{agreeR } \xi \Gamma \Gamma'$ and $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$, then there exists Γ'_1, Γ'_2 such that $\Gamma'_1 \ddagger \Gamma'_2 \ddagger \Gamma'$, and $\text{agreeR } \xi \Gamma_1 \Gamma'_1$ and $\text{agreeR } \xi \Gamma_2 \Gamma'_2$.*

Proof. By induction on the derivation of $\text{agreeR } \xi \Gamma \Gamma'$ and lemmas in Section 9.1. □

Lemma 9.5. *For context Γ, Γ' , terms m, A and mapping ξ , if there is $\Gamma \vdash m : A$ and $\text{agreeR } \xi \Gamma \Gamma'$, then there is $\Gamma' \vdash m[\xi] : A[\xi]$.*

Proof. By induction on the derivation of $\Gamma \vdash m : A$. We shall only discuss the application case in detail, as the other cases are proven by application of the induction hypothesis and the lemmas in Section 9.1.

- For the $\text{APP-U} \rightarrow$ case, Lemma 9.4 is applied to split the context Γ into two contexts Γ'_1 and Γ'_2 such that there is $\Gamma'_1 \ddagger \Gamma'_2 \ddagger \Gamma'$ and $\text{agreeR } \xi \Gamma_1 \Gamma'_1$ and $\text{agreeR } \xi \Gamma_2 \Gamma'_2$. From $|\Gamma_2|$ and Lemma 9.2 we know that there is $|\Gamma'_2|$. At this point, the induction hypothesis allows us to apply $\text{APP-U} \rightarrow$ to prove the goal.
- For the $\text{APP-L} \rightarrow$ case, Lemma 9.4 is applied to split the context Γ into two contexts Γ'_1 and Γ'_2 such that there is $\Gamma'_1 \ddagger \Gamma'_2 \ddagger \Gamma'$ and $\text{agreeR } \xi \Gamma_1 \Gamma'_1$ and $\text{agreeR } \xi \Gamma_2 \Gamma'_2$. At this point, the induction hypothesis allows us to apply $\text{APP-L} \rightarrow$ to prove the goal.
- For the $\text{APP-U} \multimap$ case, Lemma 9.4 is applied to split the context Γ into two contexts Γ'_1 and Γ'_2 such that there is $\Gamma'_1 \ddagger \Gamma'_2 \ddagger \Gamma'$ and $\text{agreeR } \xi \Gamma_1 \Gamma'_1$ and $\text{agreeR } \xi \Gamma_2 \Gamma'_2$. From $|\Gamma_2|$ and Lemma 9.2 we know that there is $|\Gamma'_2|$. At this point, the induction hypothesis allows us to apply $\text{APP-U} \multimap$ to prove the goal.
- For the $\text{APP-L} \multimap$ case, Lemma 9.4 is applied to split the context Γ into two contexts Γ'_1 and Γ'_2 such that there is $\Gamma'_1 \ddagger \Gamma'_2 \ddagger \Gamma'$ and $\text{agreeR } \xi \Gamma_1 \Gamma'_1$ and $\text{agreeR } \xi \Gamma_2 \Gamma'_2$. At this point, the induction hypothesis allows us to apply $\text{APP-L} \multimap$ to prove the goal.

□

Theorem 9.6. *Weakening is admissible for CLC variables of non-linear type. For context Γ and terms m, A, B , if there is $\Gamma \vdash m : A$, then there is $\Gamma, x :_{\mathcal{U}} B \vdash m : A$.*

Proof. Using AGREER-WK and Lemma 9.1 a proof of $\text{agreeR } id \Gamma (\Gamma, x :_{\mathcal{U}} B)$ can be constructed. Then by Lemma 9.5, the theorem can be proven. □

10 Substitution Lemmas of CLC (`clc_substitution.v`)

Similar to the proof of weakening, we first define an *agreeS* relation between two contexts Γ, Δ and a mapping σ from variables to terms.

$$\begin{array}{c}
\frac{}{\text{agreeS } \sigma \epsilon \epsilon}^{\text{AGREES-}\epsilon} \quad \frac{\text{agreeS } \sigma \Delta \Gamma \quad x \notin FV(\Delta) \cup FV(\Gamma)}{\text{agreeS } (\sigma \cup (x, x)) (\Delta, x :_U A[\sigma]) (\Gamma, x :_U A)}^{\text{AGREES-U}} \\
\\
\frac{\text{agreeS } \sigma \Delta \Gamma \quad x \notin FV(\Delta) \cup FV(\Gamma)}{\text{agreeS } (\sigma \cup (x, x)) (\Delta, x :_L A[\sigma]) (\Gamma, x :_L A)}^{\text{AGREES-L}} \\
\\
\frac{\text{agreeS } \sigma \Delta \Gamma \quad \bar{\Delta} \vdash n : A[\sigma] \quad x \notin FV(\Delta) \cup FV(\Gamma)}{\text{agreeS } (\sigma \cup (x, n)) \Delta (\Gamma, x :_U A)}^{\text{AGREES-WKU}} \\
\\
\frac{\Delta_1 \ddagger \Delta_2 \ddagger \Delta \quad \text{agreeS } \sigma \Delta_1 \Gamma \quad \Delta_2 \vdash n : A[\sigma] \quad x \notin FV(\Delta) \cup FV(\Gamma)}{\text{agreeS } (\sigma \cup (x, n)) \Delta (\Gamma, x :_L A)}^{\text{AGREES-WKL}} \\
\\
\frac{A \preceq B \quad \bar{\Delta} \vdash B[\sigma] : U_i \quad \text{agreeS } \sigma \Delta (\Gamma, x :_U A)}{\text{agreeS } \sigma \Delta (\Gamma, x :_U B)}^{\text{AGREES-CONVU}} \\
\\
\frac{A \preceq B \quad \bar{\Delta} \vdash B[\sigma] : L_i \quad \bar{\Gamma} \vdash B : L_i \quad \text{agreeS } \sigma \Delta (\Gamma, x :_L A)}{\text{agreeS } \sigma \Delta (\Gamma, x :_L B)}^{\text{AGREES-CONVL}}
\end{array}$$

10.1 Properties of *agreeS*

Lemma 10.1. *For any context Γ and identity mapping id , there is $\text{agreeS } id \Gamma \Gamma$.*

Proof. By induction on the structure of Γ . □

Lemma 10.2. *For contexts Δ, Γ and mapping σ , if there is $\text{agreeS } \sigma \Delta \Gamma$, then there is $\text{agreeS } \sigma \bar{\Delta} \bar{\Gamma}$.*

Proof. By induction on the derivation of $\text{agreeS } \sigma \Delta \Gamma$. □

10.2 Substitution Lemma

Lemma 10.3. *For contexts $\Delta, \Gamma, \Gamma_1, \Gamma_2$ and mapping σ , if there is $\text{agreeS } \sigma \Delta \Gamma$ and $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$, then there exists contexts Δ_1, Δ_2 such that $\Delta_1 \ddagger \Delta_2 \ddagger \Delta$ and $\text{agreeS } \sigma \Delta_1 \Gamma_1$ and $\text{agreeS } \sigma \Delta_2 \Gamma_2$.*

Proof. By induction on the derivation of $\text{agreeS } \sigma \Delta \Gamma$ and lemmas in Section 10.1. □

Lemma 10.4. *Generalized Substitution Lemma. For context Γ, Δ , terms m, A and mapping σ , if there is $\Gamma \vdash m : A$ and $\text{agreeS } \sigma \Delta \Gamma$, then there is $\Delta \vdash m[\sigma] : A[\sigma]$.*

Proof. The proof proceeds by induction on the derivation of $\Gamma \vdash m : A$. Similar to the proof of Lemma 9.5, the interesting cases are the application cases where Lemma 10.3 must be utilized to split the $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$ judgments for use in the induction hypothesis. □

10.3 Corollaries of Substitution

Corollary 10.4.1. *For contexts $\Gamma_1, \Gamma_2, \Gamma$ and terms A, B, m, n , if there is $\Gamma_1, x :_U A \vdash m : B$ and $|\Gamma_2|$ and $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$ and $\Gamma_2 \vdash n : A$, then there is $\Gamma \vdash m[n/x] : B[n/x]$.*

Corollary 10.4.2. *For contexts $\Gamma_1, \Gamma_2, \Gamma$ and terms A, B, m, n , if there is $\Gamma_1, x :_L A \vdash m : B$ and $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$ and $\Gamma_2 \vdash n : A$, then there is $\Gamma \vdash m[n/x] : B[n/x]$.*

Corollary 10.4.3. *For context Γ , terms m, A, B, C and natural number i , if there is $B \equiv A$ and $\bar{\Gamma} \vdash A : U_i$ and $\Gamma, x :_U A \vdash m : C$, then there is $\Gamma, x :_U B \vdash m : C$.*

Corollary 10.4.4. *For context Γ , terms m, A, B, C and natural number i , if there is $B \equiv A$ and $\bar{\Gamma} \vdash A : L_i$ and $\Gamma, x :_L A \vdash m : C$, then there is $\Gamma, x :_L B \vdash m : C$.*

11 Typing Validity of CLC (clc_validity.v)

In this section, we prove that the types of all CLC terms are themselves well-sorted.

Lemma 11.1. *For contexts $\Gamma_1, \Gamma_2, \Gamma$, if there is $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$ and $\Gamma \vdash$, then there is $\Gamma_1 \vdash$ and $\Gamma_2 \vdash$.*

Proof. By induction on the derivation of $\Gamma_1 \ddagger \Gamma_2 \ddagger \Gamma$ and the properties of $_ \ddagger _ \ddagger _$ discussed in Section 5.1. □

Theorem 11.2. *The validity of typing theorem. For any context Γ and terms m, A , if there is $\Gamma \vdash$ and $\Gamma \vdash m : A$, then there exists sort s and natural number i such that $\bar{\Gamma} \vdash A : s_i$.*

12 Subject Reduction of CLC (clc_soundness.v)

Theorem 12.1. *For any context Γ and terms m, n, A , if $\Gamma \vdash$ and $\Gamma \vdash m : A$ and $m \rightsquigarrow n$, then there is $\Gamma \vdash n : A$.*

Proof. The proof proceeds by induction on the derivation of $\Gamma \vdash m : A$. The interesting cases are the application cases which we shall discuss in detail.

- For the $\text{APP-}U \rightarrow$ case,

□