Axiomatizing Lütten & Vogler’s ready simulation for finite processes in $\text{CLL}_R$ *

Yan Zhang†, Zhaohui Zhu†, Jinjin Zhang, Yong Zhou
1 College of Computer Science
Nanjing University of Aeronautics and Astronautics
2 College of Information Science
Nanjing Audit University

February 13, 2015

Abstract

In the framework of logic labelled transition system, a variant of weak ready simulation has been presented by Lütten and Vogler. It has been shown that such behavioural preorder is the largest precongruence w.r.t parallel and conjunction composition satisfying desired properties. This paper offers a ground-complete axiomatization for this precongruence over processes containing no recursion in the calculus $\text{CLL}_R$. Compared with usual inference system for process calculus, in addition to axioms about process operators, such system contains a number of axioms to characterize the interaction between process operators and logical operators.

Keywords: process calculus, weak ready simulation, logic labelled transition system, axiomatization, $\text{CLL}_R$

1 Introduction

It is well-known that process algebra and temporal logic take different standpoint for looking at specifications and verifications of reactive and concurrent systems, and offer complementary advantages [21]. To take advantage of these two paradigms when designing systems, a few theories for heterogeneous specifications have been proposed, e.g., [4, 5, 10, 12, 15, 16, 17, 20]. Among them, Lütten and Vogler propose the notion of logic labelled transition system (Logic LTS or LLTS for short), which combines operational and logical styles of specification in one unified framework [15, 16, 17]. In particular, a variant of weak ready simulation has been presented in [16], which is adopted to capture refinement relation between processes in the presence of logical operators. It has been shown that such simulation is the largest precongruence w.r.t parallel and conjunction satisfying desired properties [16]. Moreover, in addition to usual process operators (e.g., CSP-style parallel composition, hiding, etc) and logic

---

*This work received financial support of the National Natural Science of China (No. 60973045) and Fok Ying-Tung Education Foundation.
†Corresponding author. Email: zhaohui@nuaa.edu.cn, zhaohui.nuaa@gmail.com
operators (disjunction and conjunction), some standard temporal logic operators, such as “always” and “unless”, are also integrated into this framework [17].

In a word, Lütten and Vogler offer a framework which allows one to freely mix operational and logic operators when designing systems.

Lütten and Vogler’s approach is entirely semantic, and doesn’t provide any kind of syntactic calculus. Recently, the first three authors of this paper explore recursive operations over LLTS in a pure process-algebraic style. A LLTS-oriented process calculus $\text{CLL}_R$ is presented, and the uniqueness of solutions of equations in $\text{CLL}_R$ is established under a certain circumstance [22].

It is one of important topics in concurrency theory that giving axiomatization for behaviour relations. For example, Milner gives an axiomatization for observational congruence in CCS [19]; Baeten and Bravetti extend Milner’s this work and provide an axiomatization over TCP+REC [2], where TCP+REC is a fragment of TCP+REC which is a generic process language that embodies features of the classical process algebras CCS, CSP and ACP; Lin offers complete inference systems for late and early weak bisimulation equivalences for processes without involving recursion in $\pi$-calculus [14]; Aceto et al. explore the axiomatization of weak simulation semantics systematically over BCCSP (without recursion) [1]. Although Lütten and Vogler’s original paper [16] mentions some sound laws, a complete set of axioms seems out of reach. As the main contribution of this paper we intend to provide a proof system for Lütten and Vogler’s weak ready simulation over $\text{CLL}_R$-processes with finite behaviour, and demonstrate its soundness and ground-completeness.

The rest of this paper is organized as follows. The notion of Logic LTS and the calculus $\text{CLL}_R$ are recalled in the next section. The inference system is presented in Section 3, along with the soundness proof. Section 4 demonstrates that the inference system is ground-complete for processes with finite behaviour. The paper is concluded with Section 5, where a brief discussion is given.

2 Preliminaries

The purpose of this section is to fix our notation and terminology, and to introduce some concepts that underlie our work in all other parts of the paper.

2.1 Logic LTS and ready simulation

Let $\text{Act}$ be the set of visible action names ranged over by $a, b$, etc., and let $\text{Act}_\tau$ denote $\text{Act}\cup\{\tau\}$ ranged over by $\alpha$ and $\beta$, where $\tau$ represents invisible actions. A labelled transition system with predicate is a quadruple $(P, \text{Act}_\tau, \rightarrow, F)$, where $P$ is a set of states, $\rightarrow \subseteq P \times \text{Act}_\tau \times P$ is the transition relation and $F \subseteq P$.

As usual, we write $p \overset{\alpha}{\rightarrow} F q$ (or, $p \not\overset{\alpha}{\rightarrow} F q$) if $\exists q \in P, p \overset{\alpha}{\rightarrow} q$ ($\not\exists q \in P, p \overset{\alpha}{\rightarrow} q$, resp.).

The ready set $\{\alpha \in \text{Act}_\tau | \not\exists q \in F\}$ of a given state $p$ is denoted by $\mathcal{I}(p)$. A state $p$ is stable if $p \not\overset{\tau}{\rightarrow}$. A number of useful decorated transition relations are given:

- $p \overset{\alpha}{\rightarrow}_F q$ iff $p \overset{\alpha}{\rightarrow} q$ and $p, q \not\in F$;
- $p \Rightarrow q$ iff $p(\Rightarrow)^* q$, where $(\Rightarrow)^*$ is the transitive and reflexive closure of $\Rightarrow$;
- $p \overset{\alpha}{\Rightarrow} q$ iff $\exists r, s \in P, p \overset{\alpha}{\rightarrow} r \Rightarrow s \overset{\alpha}{\rightarrow} q$;
- $p \overset{\alpha}{\Rightarrow} F q$ iff $p \overset{\alpha}{\Rightarrow} q$ with $\alpha \in \text{Act}_\tau \cup \{\epsilon\}$;
- $p \overset{\alpha}{\Rightarrow}_F q$ iff there exists a sequence of $\tau$-transitions from $p$ to $q$ such that all states along this sequence, including $p$ and $q$, are not in $F$; the decorated
transition $p \xrightarrow{\gamma} F q$ may be defined similarly:

$$p \xrightarrow{\gamma} F q \text{ iff } p \xrightarrow{\gamma} F q \not\xrightarrow{\tau} \text{ with } \gamma \in \text{Act}_\tau \cup \{\epsilon\}.$$  

Notice that the notation $p \equiv_F q$ in [16, 17] has the same meaning as $p \xrightarrow{\gamma} F q$ in this paper, while $p \equiv q$ in this paper does not involve any requirement on $F$-predicate.

**Definition 2.1** (Logic LTS [16]). An LTS $(P, \text{Act}_\tau, \rightarrow, F)$ is an LLTS if, for each $p \in P$,

- **(LTS1)** $p \in F$ if $\exists \alpha \in I(p) \forall q \in P (p \xrightarrow{\alpha} q \text{ implies } q \in F)$;
- **(LTS2)** $p \in F$ if $\exists q \in P, p \xrightarrow{\tau} F q$.

Moreover, an LTS $(P, \text{Act}_\tau, \rightarrow, F)$ is $\tau$-pure if, for each $p \in P$, $p \xrightarrow{\tau}$ implies $\exists a \in \text{Act}. p \xrightarrow{a}$.

Compared with usual LTSs, one distinguishing feature of LLTS is that it involves consideration of inconsistencies. The main motivation behind such consideration lies in dealing with inconsistencies caused by conjunctive composition. In the notion above, the predicate $F$ is used to denote the set of all inconsistent states that represent empty behaviour that cannot be implemented [17]. In the sequel, we shall use the phrase “inconsistency predicate” to refer to $F$. The condition (LTS1) formalizes the backward propagation of inconsistencies, and (LTS2) captures the intuition that divergence (i.e., infinite sequences of $\tau$-transitions) should be viewed as catastrophic. For more intuitive ideas and motivation about inconsistency, the reader may refer [15, 16].

The notion of ready simulation below is adopted to capture the refinement relation in [16, 17], which is a variant of the usual notion of weak ready simulation [5, 13]. It has been proven that such kind of ready simulation is the largest precongruence w.r.t parallel composition and conjunction which satisfies the desired property that an inconsistent specification can only be refined by inconsistent ones (see Theorem 21 in [16]).

**Definition 2.2** (Ready simulation on LLTS [16]). Let $(P, \text{Act}_\tau, \rightarrow, F)$ be a LLTS. A relation $R \subseteq P \times P$ is a stable ready simulation relation, if for any $(p, q) \in R$ and $a \in \text{Act}$

- **(RS1)** both $p$ and $q$ are stable;
- **(RS2)** $p \not\in F$ implies $q \not\in F$;
- **(RS3)** $p \xrightarrow{\tau} F p'$ implies $\exists q'. q \xrightarrow{a} F q'$ and $(p', q') \in R$;
- **(RS4)** $p \not\in F$ implies $I(p) = I(q)$.

We say that $p$ is stable ready simulated by $q$, in symbols $p \triangleright_{RS} q$, if there exists a stable ready simulation relation $R$ with $(p, q) \in R$. Further, $p$ is ready simulated by $q$, written $p \triangleright_{RS} q$, if $\forall p'(p \xrightarrow{\tau} F p' \text{ implies } \exists q'(q \xrightarrow{a} F q' \text{ and } p' \triangleright_{RS} q'))$.

The kernels of $\triangleright_{RS}$ and $\sqsubseteq_{RS}$ are denoted by $\approx_{RS}$ and $=_{RS}$ resp.. It is easy to see that $\triangleright_{RS}$ itself is a stable ready simulation relation and both $\triangleright_{RS}$ and $\sqsubseteq_{RS}$ are pre-order.

### 2.2 The calculus $\text{CLL}_R$ and its operational semantics

This subsection introduces the LLTS-oriented process calculus $\text{CLL}_R$ presented in [22]. Let $V_{AR}$ be an infinite set of variables. The terms of $\text{CLL}_R$ can be given
by the following BNF grammar

\[ t ::= 0 \mid \bot \mid (\alpha.t) \mid (t \sqcup t) \mid (t \land t) \mid (t \sqcap t) \mid (t \| t) \mid X \mid \langle Z|E \rangle \]

where \( X \in V_{AR}, \alpha \in Act_\tau, A \subseteq Act \) and recursive specification \( E = E(V) \) with \( V \subseteq V_{AR} \) is a set of equations \( \{ X = t | X \in V \} \) and \( Z \) is a variable in \( V \) that acts as the initial variable.

Most of these operators are from CCS \[18\] and CSP \[11\]: 0 is the process capable of doing no action; \( \alpha.t \) is action prefixing; \( \Box \) is non-deterministic external choice; \( ||_A \) is a CSP-style parallel composition. \( \bot \) represents an inconsistent process with empty behavior. \( \lor \) and \( \land \) are logical operators, which are intended for describing logical combinations of processes.

For any term \( \langle Z|E \rangle \) with \( E = E(V) \), each variable in \( V \) is bound with scope \( E \). This induces the notion of free occurrence of variable, bound (and free) variables and \( \alpha \)-equivalence as usual. A term \( t \) is a process if it is closed, that is, it contains no free variable. The set of all processes is denoted \( \Sigma_{CLL,\tau} \). Unless noted otherwise we use \( p, q, r \) to represent processes. Throughout this paper, as usual, we assume that recursive variables are distinct from each other and no recursive variable has free occurrence; moreover we don’t distinguish between \( \alpha \)-equivalent terms and use \( \equiv \) for both syntactical identical and \( \alpha \)-equivalence. In the sequel, we often denote \( \langle X|\{ X = t_X \} \rangle \) briefly by \( \langle X|X = t_X \rangle \).

For any recursive specification \( E(V) \) and term \( t \), the term \( \langle t|E \rangle \) is obtained from \( t \) by simultaneously replacing all free occurrences of each \( X(e \in V) \) by \( \langle X|E \rangle \), that is, \( \langle t|E \rangle \equiv t\langle X|E \rangle/X : X \in V \). For example, consider \( t \equiv X \Box a.\langle Y|X = X \lor Y \rangle \) and \( E(\{ X \}) = \{ X = t_X \} \) then \( \langle t|E \rangle \equiv \langle X|X = t_X \rangle \Box a.\langle Y|X = X \lor t_X \rangle \lor Y \). In particular, for any \( E(V) \) and \( t \equiv X, \langle t|E \rangle \equiv \langle X|E \rangle \) whenever \( X \in V \) and \( \langle t|E \rangle \equiv X \) if \( X \notin V \).

An occurrence of \( X \) in \( t \) is strongly (or, weakly) guarded if such occurrence is within some subexpression \( a.t_1 \) with \( a \in Act \) (\( \tau.t_1 \) or \( t_1 \lor t_2 \) resp.). A variable \( X \) is strongly (or, weakly) guarded in \( t \) if each occurrence of \( X \) is strongly (weakly resp.) guarded. A recursive specification \( E(V) \) is guarded if for each \( X \in V \) and \( Z = t_Z \in E(V) \), each occurrence of \( X \) in \( t_Z \) is (weakly or strongly) guarded. As usual, we assume that all recursive specifications considered in the remainder of this paper are guarded.

SOS rules of \( \text{CLL}_R \) are listed in Table \[1\] where \( a \in Act, \alpha \in Act_\tau \) and \( A \subseteq Act \). All rules are divided into two parts:

Operational rules specify behaviours of processes. Negative premises in Rules \( Ra_2, Ra_3, Ra_{13} \) and \( Ra_{14} \) give \( \tau \)-transition precedence over visible transitions, which guarantees that the transition model of \( \text{CLL}_R \) is \( \tau \)-pure. Rules \( Ra_9 \) and \( Ra_{10} \) illustrate that the operational aspect of \( t_1 \lor t_2 \) is same as internal choice in usual process calculus. Rule \( Ra_6 \) reflects that conjunction operator is a synchronic product for visible transitions. The operational rules of the other operators are as usual.

Predicate rules specify the inconsistency predicate \( F \). Rule \( Rp_1 \) says that \( \bot \) is inconsistent. Hence \( \bot \) cannot be implemented. While 0 is consistent and implementable. Thus 0 and \( \bot \) represent different processes. Rule \( Rp_3 \) reflects that if both two disjunctive parts are inconsistent then so is the disjunction. Rules \( Rp_4 - Rp_9 \) describe the system design strategy that if one part is inconsistent, then so is the whole composition. Rules \( Rp_{10} \) and \( Rp_{11} \) reveal that
a stable conjunction is inconsistent whenever its conjuncts have distinct ready sets. Rules $Rp_{13}$ and $Rp_{15}$ are used to capture (LTS2) in Def. 2.11. Intuitively, these two rules say that if all stable $\tau$-descendants of $z$ are inconsistent, then $z$ itself is inconsistent.

### Operational rules

| Rule No. | Description |
|----------|-------------|
| $Ra_1$  | $\alpha \cdot x_1 \leadsto x_1$ |
| $Ra_2$  | $x_1 \leadsto y_1$ |
| $Ra_3$  | $x_1 \leadsto y_2$ |
| $Ra_4$  | $x_1 \leadsto y_1$ |
| $Ra_5$  | $x_1 \leadsto y_2$ |
| $Ra_6$  | $x_1 \leadsto y_1 \land y_2$ |
| $Ra_7$  | $x_1 \land x_2 \leadsto x_1 \land x_2$ |
| $Ra_8$  | $x_1 \land x_2 \leadsto x_1 \land y_2$ |
| $Ra_9$  | $x_1 \lor x_2 \leadsto x_1 \lor x_2$ |
| $Ra_{10}$ | $x_1 \lor x_2 \leadsto y_1 \land x_2$ |
| $Ra_{11}$ | $x_1 \land x_2 \leadsto y_1 \land y_2$ |
| $Ra_{12}$ | $x_1 \land x_2 \leadsto y_1 \land y_2$ |
| $Ra_{13}$ | $x_1 \land x_2 \leadsto y_1 \land y_2$ |
| $Ra_{14}$ | $x_1 \land x_2 \leadsto y_1 \land y_2$ |

### Predicative rules

| Rule No. | Description |
|----------|-------------|
| $Rp_{1}$ | $(\alpha, x_1 \rightarrow y_1)$ |
| $Rp_{2}$ | $(x_1 \rightarrow y_1)$ |
| $Rp_{3}$ | $(x_1 \rightarrow y_2)$ |
| $Rp_{4}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{5}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{6}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{7}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{8}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{9}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{10}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{11}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{12}$ | $(x_1 \rightarrow y_1 \land y_2)$ |
| $Rp_{13}$ | $(x_1 \rightarrow y_1 \land y_2)$ |

Table 1: SOS rules of $CLL_R$

It has been shown that $CLL_R$ has the unique stable transition model $M_{CLL_R}$ [22], which exactly consists of all positive literals of the form $t \rightarrow t'$ or $t \rightarrow F$ that are provable in $\text{Strip}(CLL_R, M_{CLL_R})$. Here $\text{Strip}(CLL_R, M_{CLL_R})$ is the stripped version of $CLL_R$ w.r.t $M_{CLL_R}$. Each rule in $\text{Strip}(CLL_R, M_{CLL_R})$ is of the form $\text{nprem}(r)$ for some ground instance $r$ of rules in $CLL_R$ such that $M_{CLL_R} \models nprem(r)$, where $\text{nprem}(r)$ (or, $\text{pprem}(r)$) is the set of negative (positive resp.) premises of $r$, $\text{conc}(r)$ is the conclusion of $r$ and $M_{CLL_R} \models nprem(r)$ means that for each $t \notin nprem(r)$, $t \leadsto s \notin M_{CLL_R}$ for any $s \in T(\Sigma_{CLL_R})$.

The LTS associated with $CLL_R$, in symbols $LTS(CLL_R)$, is the quadruple $(T(\Sigma_{CLL_R}), \text{Act}_\tau, \rightarrow_{CLL_R}, F_{CLL_R})$, where $p \rightarrow_{CLL_R} p'$ iff $p \leadsto p' \in M_{CLL_R}$.
and \( p \in F_{\text{CLL}_R} \) iff \( pF \in M_{\text{CLL}_R} \). Therefore \( p^\alpha_{\text{CLL}_R} p' \) (or, \( p \in F_{\text{CLL}_R} \)) iff 
\[ \text{Strip}(\text{CLL}_R, M_{\text{CLL}_R}) \vdash p^\alpha_{\text{CLL}_R} p' \] (resp.) for any \( p, p' \) and \( \alpha \in \text{Act}_\tau \). For 
simplification, in the following we omit the subscripts in \( \alpha_{\text{CLL}_R} \) and \( F_{\text{CLL}_R} \).

We end this section by quoting some results from [22].

**Lemma 2.3.** Let \( p \) and \( q \) be any two processes. Then
1. \( p \lor q \in F \) iff \( p, q \in F \);
2. \( \alpha.p \in F \) iff \( p \in F \) for each \( \alpha \in \text{Act}_\tau \);
3. \( p \otimes q \in F \) iff either \( p \in F \) or \( q \in F \) with \( \otimes \in \{ \Box, || \} \);
4. \( p \otimes q \in F \) implies \( p \land q \in F \);
5. \( 0 \not\in F \) and \( \bot \in F \).

**Theorem 2.4.** \( \text{LTS}(\text{CLL}_R) \) is a \( \tau \)-pure LLTS. Moreover if \( p \in F \) and \( \tau \in I(p) \) 
then \( \forall q(p \Rightarrow q \implies q \in F) \).

**Theorem 2.5** (precongruence). If \( p \sqsubseteq_{RS} q \) then \( C_X\{p/X\} \sqsubseteq_{RS} C_X\{q/X\} \), 
where \( C_X \) is any context defined as usual.

3 **Axiomatic system \( AX_{\text{CLL}} \) and its soundness**

This section is devoted to formulating an axiomatic system for the precongruence \( \sqsubseteq_{RS} \) and proving its soundness. For the moment, we don’t know whether a ground-complete proof system exists for the full calculus CLL\(_R\). This paper will restrict itself to the finite fragment, i.e., leave out recursive operator.

3.1 **\( AX_{\text{CLL}} \)**

Since inconsistency predicate \( F \) (more precisely, \( F_{\text{CLL}_R} \)) is involved in the definition of \( \sqsubseteq_{RS} \), it could be expected that some algebraic laws hold only for processes satisfying certain conditions concerning consistency. However, since \( F \) itself is in semantic category, it is illegal that formulating these conditions in terms of \( F \) in axiomatic systems. Therefore, in order to introduce the axiomatic system \( AX_{\text{CLL}} \), a few preliminary definitions are given below, which are needed to express side conditions of some axioms.

**Definition 3.1** (Basic Process Term). The basic process terms are defined by
\[
\text{BNF} \quad t ::= 0 \mid (\alpha.t) \mid (t \lor t) \mid (t \otimes t) \mid (t ||_A t), \quad \alpha \in \text{Act}_\tau \text{ and } A \subseteq \text{Act}.
\]
We denote \( T(\Sigma_B) \) as the set of all basic process terms.

At a later stage, we will see that the set \( T(\Sigma_B) \) is sufficiently expressive to describe all consistent processes with finite behaviours modulo \( =_{RS} \). Moreover, through referring \( T(\Sigma_B) \), we can formulate syntactically algebraic laws that hold conditionally, e.g., Axioms D4 and EXP2.

**Remark 3.2.** Since all proofs in this section does not depend on the finiteness of processes’ behaviour, all results given in this section are still valid if we extend \( T(\Sigma_B) \) by adding the item \( \langle X|E \rangle \) in BNF above, where \( \langle X|E \rangle \) is any strongly guarded processes in \( T(\Sigma_{\text{CLL}_R}) \) in which neither conjunction operator nor \( \bot \) occurs. We denote \( ET(\Sigma_B) \) as the set of all process terms generating by such extended BNF. For the purpose of this paper \( T(\Sigma_B) \) is sufficient.
By Lemma \ref{lem:consistency_preservation}, it is easy to see that the operators $\alpha(\cdot), \lor, \Box$ and $\parallel A$ preserve consistency. Thus an immediate consequence of Lemma \ref{lem:consistency_preservation} is

\begin{lemma}
\label{lem:initial_empty_set}
T(\Sigma_B) \cap F = \emptyset.
\end{lemma}

Let $< t_0, t_1, \ldots, t_{n-1} >$ be a finite sequence of process terms with $n \geq 0$. We define the general external choice $\Box t_i$ by recursion:

$$\Box t_i \triangleq 0, \quad \Box t_i \triangleq t_0, \quad \text{and} \quad \Box t_i \triangleq (\Box t_i) \bullet t_k \text{ for } k \geq 1.$$  

Moreover, given a finite sequence $< t_0, \ldots, t_{n-1} >$ and $S \subseteq \{ t_0, \ldots, t_{n-1} \}$, the general external choice $\Box S$ is defined as $\Box S \triangleq \Box t'_j$ where the sequence $< t'_0, \ldots, t'_{|S|} >$ is the restriction of $< t_0, \ldots, t_{n-1} >$ to $S$. In fact, up to $\equiv_{RS}$ (or, $\equiv$, see below), the order and grouping of terms in $\Box t_i$ may be ignored by virtue of commutative and associative laws of $\Box$ w.r.t $\equiv_{RS}$ (axioms EC\textsubscript{1} and EC\textsubscript{2} below, resp.).

\begin{definition}[Injective in Prefixes]
\label{def:injective_prefixes}
A process $\Box_{i<n} \alpha_i \cdot t_i$ is injective in prefixes if $\alpha_i \neq \alpha_j$ for any $i \neq j < n$.
\end{definition}

The axiomatic system $AX_{CLL}$ is reported in Table \ref{tab:axioms_CLL}. It is an inequational logic where $t = t'$ means $t \leq t'$ and $t' \leq t$. Axioms in $AX_{CLL}$ may be divided into two groups:

First the ones that involve only a single operator, which capture fundamental properties of operators, e.g., commutativity, associativity, idempotent, etc. These axioms are standard.

Second the ones that characterize the interaction between operators. Among them, the axioms $DS1$, $DS3$, $DS4$ and $ECCi(1 \leq i \leq 3)$ describe the interaction between logical and operational operators. As mentioned early, it is one distinguishing feature of LLTS that it involves consideration of inconsistencies. A number of axioms in this group embody such feature. In particular, as a consequence of considering inconsistency, side conditions are associated with $DS4$, $ECC3$ and $EXP2$. In the next subsection, we will show that these side conditions are necessary by giving counterexamples.

It should be pointed out that some axioms have been considered by Lüt teng and Vogler semantically in \cite{16}, including $DS2$, $CO2$, $CO3$ and $DIi(3 \leq i \leq 5)$.

Given the axioms and rules of inference, we assume that the resulting notions of proof, length of proof and theorem are already familiar to the reader. Following standard usage, $t \vdash t'$ means that $t \leq t'$ is a theorem of $AX_{CLL}$.

\subsection{Soundness}

This subsection will establish the soundness of $AX_{CLL}$ w.r.t $\sqsubseteq_{RS}$. Although $AX_{CLL}$ is a proof system for $CLL_{R}$-processes with finite behaviours, it is sound for the full calculus. Therefore this subsection doesn’t restrict itself to finite terms.

As usual, in order to get soundness, we need to check that all ground instances of axioms are sound w.r.t $\sqsubseteq_{RS}$ and all inference are sound. The latter immediately follows from reflexivity and transitivity of $\sqsubseteq_{RS}$ and Theorem \ref{thm:reflexivity}.
Axioms

EC1 $x \Box y = y \Box x$   DI1 $x \lor y = y \lor x$
EC2 $(x \Box y) \Box z = x \Box (y \Box z)$   DI2 $x \lor (y \lor z) = (x \lor y) \lor z$
EC3 $x \Box x = x$   DI3 $x \lor x = x$
EC4 $x \Box 0 = x$   DI4 $x \lor \perp = x$
EC5 $x \Box \perp = \perp$   DI5 $x \leq x \lor y$
CO1 $x \land y = y \land x$   DS1 $x \Box (y \lor z) \leq (x \Box y) \lor (x \Box z)$
CO2 $x \land x = x$   DS2 $x \land (y \lor z) \leq (x \land y) \lor (x \land z)$
CO3 $x \land \perp = \perp$   DS3 $x \Box (y \lor z) \leq (x \Box y) \lor (x \Box z)$
PR1 $a \land \perp = \perp$   DS4 $a \Box (x \lor y) \leq a \Box x \lor a \Box y$, where $x, y \in T(\Sigma_B)$
PR2 $\tau.x = x$   PA1 $x \Box A y = y \Box_A x$
PA2 $x \Box_A \perp = \perp$

ECC1 $\Box a_i.x_i \land \Box b_j.y_j = \perp$ if $\{a_i | i < n\} \neq \{b_j | j < m\}$
ECC2 $\Box a_i.(x_i \land y_i) \leq \Box a_i.x_i \land \Box a_i.y_i$
ECC3 $\Box a_i.x_i \land \Box a_i.y_i \leq \Box a_i.(x_i \land y_i)$ if $\Box a_i.x_i$ is injective in prefixes

EXP1

\[
\begin{align*}
\Box a_i.x_i \Box_A j < m b_j.y_j & \leq \\
\left( \begin{array}{c}
\Box a_i.(x_i \Box_A j < m b_j.y_j) \Box a_i.(x_i \Box_A y_j) \\
\Box a_i.(x_i \Box_A j < m b_j.y_j) \Box a_i.(x_i \Box_A y_j)
\end{array} \right)
\end{align*}
\]

EXP2

\[
\begin{align*}
\begin{array}{c}
\Box a_i.(x_i \Box_A j < m b_j.y_j) \Box a_i.(x_i \Box_A y_j)
\end{array}
\leq \begin{array}{c}
\Box a_i.x_i \Box_A j < m b_j.y_j, \text{ where } x_i, y_j \in T(\Sigma_B) \text{ for each } i < n \text{ and } j < m
\end{array}
\end{align*}
\]

Inference rules

| Rule   | Premise 1 | Premise 2 |
|--------|-----------|-----------|
| REF    | $t \leq t'$ | $t \leq t''$ |
| TRANS  | $t \leq t''$ | for each n-ary operator $f$ |
| CONTEXT | $t_1 \leq t'_1, \ldots, t_n \leq t'_n$ | $f(t_1, \ldots, t_n) \leq f(t'_1, \ldots, t'_n)$ |

Table 2: Axioms and inference rules of $AX_{\text{CLL}}$
Therefore the remainder of this subsection will devote itself to verifying the soundness of axioms.

We begin by giving a simple but useful property about combined processes $p \odot q$ with $\odot \in \{\sqcup, \|A, \land\}$. Roughly speaking, it says that consistent and stable $s$-derivatives of $p \odot q$ must be compositions of consistent and stable $s$-derivatives of $p$ and $q$, and the converse also (almost) holds.

**Lemma 3.5.** (1) For any $\odot \in \{\sqcup, \|A, \land\}$, if $p_1 \odot p_2 \Rightarrow_F |p_3$ then $p_1 \Rightarrow_F |p'_1$, $p_2 \Rightarrow_F |p'_2$ and $p_3 \equiv p'_1 \odot p'_2$ for some $p'_1, p'_2$.

(2) If $p_1 \Rightarrow_F |p'_1$ and $p_2 \Rightarrow_F |p'_2$ then $p_1 \odot p_2 \Rightarrow_F |p'_1 \odot p'_2$ for $\odot \in \{\sqcup, \|A\}$, and $p_1 \land p_2 \Rightarrow_F |p'_1 \land p'_2$ if $p'_1 \land p'_2 \notin F$.

**Proof.** Straightforward by applying Theorem 2.4 and Lemma 2.3.

**Lemma 3.6.** (1) For any $\odot \in \{\sqcup, \|A, \land\}$, if $p \subseteq RS p_i \land p_2$ for $i = 1, 2$.

(2) If $p \subseteq RS p_1$ and $p_2 \subseteq RS p_3$ then $p_1 \lor p_2 \subseteq RS p_3$.

(3) $p_i \land p_2 \subseteq RS p_i$ for $i = 1, 2$.

(4) If $p \subseteq RS p_2$ and $p \subseteq RS p_3$, then $p \subseteq RS p_2 \land p_3$.

**Proof.** (1,2) Straightforward.

(3) Assume $p_1 \land p_2 \Rightarrow_F |p_1 \lor p_2$. By Lemma 3.5, $p_1 \Rightarrow_F |p'_1$ and $p_1 \equiv p'_1 \land p'_2$ for some $p'_1, p'_2$. Then it suffices to show $p'_1 \land p'_2 \subseteq RS p_1$. To this end, put $R \triangleq \{(s \land t, s) | s$ and $t$ are stable}. It is routine to verify that $R$ is a stable relation, as desired.

(4) It immediately follows from Lemma 3.5 and the fact that $p \subseteq RS q$ and $p \subseteq r$ implies $p \subseteq RS q \land r$ (see [22, Lemma 4.5]).

As an immediate consequence of items (3) and (4) in previous lemma, the property below is given, which is obtained in [16].

$$p \subseteq RS p_2 \land p_3 \iff p \subseteq RS p_2 \land p_1 \subseteq RS p_3.$$  

(FP)

As pointed out by Lüttingen and Vogler [15, 16], this is a fundamental property of redy simulation in the presence of logic operators. Intuitively, it says that $p_1$ is an implementation of the specification $p_2 \land p_3$ if and only if $p_1$ implements both $p_2$ and $p_3$. Moreover, by Lemma 3.6 it is easy to see that the following equation holds.

$$p \land (p \lor q) = RS p = RS p \lor (p \land q)$$  

(Absorption)

More fundamental algebraic laws are collected in the next proposition.

**Proposition 3.7.**

(1) Commutativity: $p_1 \odot p_2 = RS p_2 \odot p_1$ for each $\odot \in \{\sqcup, \|A, \land\}$;

(2) Associativity: $(p_1 \odot p_2) \odot p_3 = RS p_1 \odot (p_2 \odot p_3)$ for each $\odot \in \{\sqcup, \|A, \land\}$;

(3) Idempotency: $p \odot p = RS p$ for each $\odot \in \{\sqcup, \|A, \land\}$;

(4) Unit element: $p \odot 0 = RS p$, $p \lor \bot = RS p$;

(5) Zero element: $p \odot \bot = RS \bot$ for each $\odot \in \{\sqcup, \|A, \land\}$;

(6) Identity property: $p \odot p = RS p$, $p \land p = RS p$. 

9
Proof. We give the proof only for Commutativity laws, the other laws are left to
the reader. Clearly Commutativity laws for \( \land \) and \( \lor \) are implied by Lemma 3.6
For \( \circ \in \{ \circ, \cdot, \| \}_A \), the argument is similar to that in the proof of Lemma 3.6(3),
that is, by Lemma 3.5 it is enough to check that the relation \( R \) below is a stable ready simulation relation.

\[
R = \{(p \circ q, q, p) : p, q \text{ are stable}\} \cup \text{Id}
\]

Remark 3.8. Due to Commutativity, Associativity, Idempotency and Absorption
laws of \( \land \) and \( \lor \), modulo \( \equiv_{RS} \), the structure \( T(\Sigma_{\text{CLL}}) \), \( \land, \lor, \equiv \) is a
lattice. In fact, such lattice is distributive by Prop. 3.10 given later. Moreover,
by Lemma 3.6(3) and (FP), the partial order corresponding to the lattice
\( T(\Sigma_{\text{CLL}}) \), \( \land, \lor \) is indeed is \( \equiv_{RS} \), that is, \( p \equiv_{RS} q \) iff \( p \land q \equiv_{RS} p \) for any
\( p, q \in T(\Sigma_{\text{CLL}}) \).

In the following, we shall deal with a few of laws referring to different operators
in one (in)equation. In order to show so-called distributive law, the next lemma is needed which reveals that there exist “canonical” evolving paths from
\( p \circ (p_2 \lor p_3) \) to its stable \( \varepsilon \)-derivatives (if exist).

Lemma 3.9. Let \( \circ \in \{ \circ, \\land, \lor \}_A \). If \( p_1 \circ (p_2 \lor p_3) \iff_F |p_4 \) then there are \( p'_1 \)
and \( r_i (i \leq n \text{ and } n > 0) \) such that (1) \( p_1 \circ (p_2 \lor p_3) \equiv r_0 \iff_F, \ldots, \iff_F r_n \equiv p_4 \),
(2) \( p_1 \iff p'_1 \), (3) \( r_j \equiv p'_1 \circ (p_2 \lor p_3) \) and \( r_{j+1} \equiv p'_1 \circ p_k \) for some \( j < n \text{ and } k \in \{2, 3\} \).

Proof. Since \( p_1 \circ (p_2 \lor p_3) \iff p_4 \) and \( p_2 \lor p_3 \iff, p_1 \circ (p_2 \lor p_3) (\iff_F)^m |p_4 \) for
some \( m > 0 \). The rest of the proof is routine by induction on \( m \).

The following Distributive law with \( \circ = \land \) was first proved in [16].

Proposition 3.10 (Distributive). \( p_1 \circ (p_2 \lor p_3) \equiv_{RS} (p_1 \circ p_2) \lor (p_1 \circ p_3) \) for
each \( \circ \in \{ \circ, \\land, \lor \}_A \).

Proof. The inequation \( (p_1 \circ p_2) \lor (p_1 \circ p_3) \equiv_{RS} p_1 \circ (p_2 \lor p_3) \) immediately follows from
Theorem 2.5 and Lemma 3.6(1)(2). For the converse inequation, suppose
\( p_1 \circ (p_2 \lor p_3) \iff_F |p_4 \). Then by Theorem 2.4 and Lemma 3.9 it is easy to get
\( (p_1 \circ p_2) \lor (p_1 \circ p_3) \iff_F |p_4 \). Hence \( p_1 \circ (p_2 \lor p_3) \equiv_{RS} (p_1 \circ p_2) \lor (p_1 \circ p_3) \).

Since \( T(\Sigma_{\text{CLL}}) \), \( \land, \lor \) is a lattice, it immediately follows from Prop. 3.10
with \( \circ = \land \) that \( p_1 \lor (p_2 \land p_3) \equiv_{RS} (p_1 \lor p_2) \land (p_1 \lor p_3) \).

Proposition 3.11. \( \alpha.p_1 \circ \alpha.p_2 \equiv_{RS} \alpha.(p_1 \lor p_2) \) for each \( \alpha \in \text{Act}_\varepsilon \).

Proof. \( p_1 \equiv_{RS} p_1 \lor p_2 \) and \( p_2 \equiv_{RS} p_1 \lor p_2 \) (by Lemma 3.5(1))
\( \Rightarrow \alpha.p_1 \equiv_{RS} \alpha.(p_1 \lor p_2) \) and \( \alpha.p_2 \equiv_{RS} \alpha.(p_1 \lor p_2) \) (by Theorem 2.5)
\( \Rightarrow \alpha.p_1 \circ \alpha.p_2 \equiv_{RS} \alpha.(p_1 \lor p_2) \) (by Theorem 2.5 and Prop. 3.7).

A natural problem arises at this point, that is, whether the inequation below holds

\[
\alpha.(p_1 \lor p_2) \equiv_{RS} \alpha.p_1 \circ \alpha.p_2.
\]

(DS)

The answer is negative by considering \( p_1 \equiv \bot \) and \( p_2 \equiv 0 \). By Lemma 2.5
\( \alpha.(\bot \lor 0) \notin F \) and \( \alpha.\bot \circ 0 \in F. \) Hence \( \alpha.(\bot \lor 0) \equiv_{RS} \alpha.\bot \circ 0. \) However
we can give a necessary and sufficient condition for the inequation (DS) with \( \alpha \in \text{Act} \) to be true. To this end, we introduce the notion

**Definition 3.12** (Uniform w.r.t \( F \)). Two processes \( p \) and \( q \) are uniform w.r.t \( F \) iff \( p \in F \) iff \( q \in F \).

**Proposition 3.13.** For each \( a \in \text{Act} \), \( a.(p_1 \lor p_2) \subseteq_{RS} a.p_1 \sqcap a.p_2 \) iff \( p_1 \) and \( p_2 \) are uniform w.r.t \( F \).

**Proof.** (Left implies Right) Suppose \( p_1 \) and \( p_2 \) are not uniform w.r.t \( F \). W.l.o.g., assume that \( p_1 \in F \) and \( p_2 \notin F \). By Lemma 2.3, we get \( a.(p_1 \lor p_2) \notin F \) and \( a.p_1 \sqcap a.p_2 \notin F \). Hence \( a.(p_1 \lor p_2) \not\subseteq_{RS} a.p_1 \sqcap a.p_2 \).

(Right implies Left) Since \( a \in \text{Act} \), it suffices to prove \( a.(p_1 \lor p_2) \not\subseteq_{RS} a.p_1 \sqcap a.p_2 \). Put

\[ R \triangleq \{(a.(p_1 \lor p_2), a.p_1 \sqcap a.p_2)\} \cup \text{Id}. \]

We will show that \( R \) is a stable ready simulation relation. It is obvious that (RS1-4) hold for each pair in \( \text{Id} \). In the following, we deal with the pair \((a.(p_1 \lor p_2), a.p_1 \sqcap a.p_2)\). Clearly, such pair satisfies (RS1) and (RS4).

(RS2) Suppose \( a.p_1 \sqcap a.p_2 \in F \). By Lemma 2.3, \( p_1 \in F \) for some \( i \in \{1, 2\} \). Then, since \( p_1 \) and \( p_2 \) are uniform w.r.t \( F \), we get \( p_1, p_2 \in F \). So \( a.(p_1 \lor p_2) \in F \).

(RS3) Suppose \( a.(p_1 \lor p_2) \not\Rightarrow_F r \). It is easy to see that \( a.p_1 \sqcap a.p_2 \not\Rightarrow_F r \). Moreover, \( a.p_1 \sqcap a.p_2 \notin F \) by \( a.(p_1 \lor p_2) \notin F \) and (RS2). So \( a.p_1 \sqcap a.p_2 \not\Rightarrow_F r \).

Notice that the situation is different if \( \alpha = \tau \). In such case, the inequation (DS) does not always hold even if \( p_1 \) and \( p_2 \) are uniform w.r.t \( F \). As a simple example, consider \( p_1 \equiv a.0 \) and \( p_2 \equiv b.0 \) with \( a \neq b \). Clearly, they are uniform w.r.t \( F \) because of \( p_1, p_2 \notin F \). Moreover, \( \tau.(a.0 \lor b.0) \not\Rightarrow_F a.0 \), and \( a.0 \sqcap b.0 \) is the unique process such that \( \tau.a.0 \sqcap \tau.b.0 \not\Rightarrow_F a.0 \sqcap b.0 \). But \( a.0 \not\subseteq_{RS} a.0 \sqcap b.0 \) due to \( a.0 \notin F \) and \( \mathcal{I}(a.0) \neq \mathcal{I}(a.0 \sqcap b.0) \). Thus \( \tau.(a.0 \lor b.0) \not\subseteq_{RS} \tau.a.0 \sqcap \tau.b.0 \).

Given the key role that general external choice \( i \downarrow p_i \) plays in the axiomatic system \( \text{AX}_{\text{CLL}} \), we need to discuss this operator in some detail. We begin with giving the following simple result, of which we omit the straightforward proof.

**Lemma 3.14.** Let \( n \geq 0 \) and \( \{a_i | i < n\} \subseteq \text{Act} \).

1. \( \square_{i \leq n} p_i \in F \) iff \( p_k \in F \) for some \( k < n \).
2. \( \square_{i \leq n} a_i.p_i \Rightarrow a_i \) for each \( i < n \).
3. If \( \square_{i \leq n} a_i.p_i \Rightarrow s \) then \( \alpha = a_k \) and \( s \equiv p_k \) for some \( k < n \).

**Proposition 3.15.** Let \( a_i, b_j \in \text{Act} \) for each \( i < n \) and \( j < m \).

1. If \( \{a_i | i < n\} \neq \{b_j | j < m\} \) then \( \square_{i \leq n} a_i.p_i \land \square_{j \leq m} b_j.q_j \not\subseteq_{RS} \perp \).
2. \( \square_{i \leq n} a_i.p_i \land \square_{j \leq m} b_j.q_j \not\subseteq_{RS} \square_{i \leq n} a_i.q_i \).

**Proof.** (1) By Rules \( Rp_{10} \) and \( Rp_{11} \), it holds trivially.

(2) If \( n = 0 \), it is trivial because of the definition of general external choice. Next we treat the case \( n > 0 \). By Lemma 3.6 and Theorem 2.7, \( a_i.(p_i \land q_i) \subseteq_{RS} a_i.p_i \) for each \( i < n \). Then \( \square_{i \leq n} a_i.(p_i \land q_i) \subseteq_{RS} \square_{i \leq n} a_i.p_i \) by Theorem 2.3 and Prop. 3.7.

Similarly, we also have \( \square_{i \leq n} a_i.(p_i \land q_i) \subseteq_{RS} \square_{i \leq n} a_i.q_i \). Hence \( \square_{i \leq n} a_i.(p_i \land q_i) \subseteq_{RS} \square_{i \leq n} a_i.p_i \land \square_{i \leq n} a_i.q_i \) by Lemma 3.6.
In the following, we provide an example to illustrate that it does not always hold that \( \square a_i, p_i \land \square a_i, q_i \not\subseteq_{RS} \square a_i, (p_i \land q_i) \).

**Example 3.16.** Consider process \( a_0, p_0 \equiv a.b.0, a_1, p_1 \equiv a.c.0, a_0, q_0 \equiv a.b.0 \) and \( a_1, q_1 \equiv a.b.0 \) where \( c \neq b \). Then, \( \square a_1, p_1 \equiv a.b.0 \land a.c.0, \square a_1, q_1 \equiv a.b.0 \land a.b.0 \) and \( \square a_i, (p_i \land q_i) \equiv a.(b.0 \land b.0) \land a.(c.0 \land b.0) \). Assume for contradiction that

\[ \square a_1, p_i \land \square a_1, q_i \not\subseteq_{RS} \square a_i, (p_i \land q_i) \].

Thus \( \square a_1, p_i \land \square a_1, q_i \not\subseteq_{RS} \square a_i, (p_i \land q_i) \). Assume for contradiction that

\( \square a_i, (p_i \land q_i) \in \text{ Strip}(\text{CLL}_R, \text{MCLL}_R) \) and \( \square a_i, (p_i \land q_i) \in \text{ F}_i \). Since \( \square a_i, p_i \not\subseteq F_i \) and \( \square a_i, q_i \not\subseteq F_i \) and \( \text{I}(\square a_i, p_i) = \text{I}(\square a_i, q_i) \), the last rule applied in the proof tree of \( \text{Strip}(\text{CLL}_R, \text{MCLL}_R) \) is of the form

\[
\begin{array}{c}
\{ sF : \square a_i, p_i \land \square a_i, q_i \Rightarrow \{ s \} \\
\square a_i, p_i \land \square a_i, q_i F
\end{array}
\]

However, since \( b.0 \land b.0 \) is an \( a \)-derivative of \( \square a_i, p_i \land \square a_i, q_i \) and \( b.0 \land b.0 \not\subseteq F_i \), the former is impossible. Moreover, since \( \square a_i, p_i \land \square a_i, q_i \) is the unique stable \( a \)-derivative of itself, the latter is also impossible due to the well-foundedness of proof tree. Thus a contradiction arises, as desired.

However, for any \( \square a_i, p_i \) with distinct prefixes, we have

**Proposition 3.17.** Let \( a_i \in \text{Act} \) for each \( i < n \). If \( \square a_i, p_i \) is injective in prefixes then \( \square a_i, p_i \land \square a_i, q_i \subseteq_{RS} \square a_i, (p_i \land q_i) \).

**Proof.** We examine the case \( n > 0 \). Since \( \{ a_i : i < n \} \subseteq \text{Act} \), it suffices to prove \( \square a_i, p_i \land \square a_i, q_i \subseteq_{RS} \square a_i, (p_i \land q_i) \). Put

\[ R \triangleq \{ (\square a_i, p_i \land \square a_i, q_i, \square a_i, (p_i \land q_i)) \} \cup \text{ Id}. \]

We need to check that \( (\square a_i, p_i \land \square a_i, q_i, \square a_i, (p_i \land q_i)) \) satisfies (RS1, 4). For the conditions (RS1, 4), it is trivial and omitted.

**(RS2)** Suppose \( \square a_i, (p_i \land q_i) \in F_i \). Then, by Lemma 3.13 \( p_k \land q_k \in F_i \) for some \( k \). Since both \( \square a_i, p_i \) and \( \square a_i, q_i \) are injective in prefixes, \( p_k \land q_k \) is the unique \( a_k \)-derivative of \( \square a_i, p_i \land \square a_i, q_i \). Therefore \( \square a_i, p_i \land \square a_i, q_i \in F_i \) comes from \( p_k \land q_k \) in \( F_i \) by Theorem 2.13 and (LTS1) in Def. 2.1 as desired.

**(RS3)** Suppose \( \square a_i, p_i \land \square a_i, q_i \Rightarrow F_i \) \( p' \). Then \( \square a_i, p_i \land \square a_i, q_i \Rightarrow F_i \) \( p' \Rightarrow F_i \) \( p' \) for some \( p'' \). Since \( \square a_i, p_i \) and \( \square a_i, q_i \) are injective in prefixes, there exists \( k < n \) such that \( \square a_i, p_i \Rightarrow p_k, \square a_i, q_i \Rightarrow q_k, a = a_k \) and \( p'' \equiv p_k \land q_k \). Clearly \( \square a_i, (p_i \land q_i) \Rightarrow p_k \land q_k \). Moreover \( \square a_i, (p_i \land q_i) \not\subseteq F_i \) by
\( \Box a_i p_i \land \Box a_i q_i \not\in F \) and (RS2). Hence \( \Box a_i (p_i \land q_i) \overset{a_i}{\rightarrow}_F p_i \land q_i \equiv p'' \overset{\tau}{\rightarrow}_F |p'| \) and \( (p', p') \in R \).

The next two propositions state the properties of the interaction of general external choice and parallel operator, which are analogous to the expansion law in usual process calculi, e.g., [13].

**Proposition 3.18.** Let \( n \geq 0, m \geq 0, A \subseteq \Act \) and \( a_i, b_j \in \Act \) for each \( i < n \) and \( j < m \).

Then

\[
\Box a_i p_i \parallel A \Box b_j q_j \sqsubseteq \text{RS} (([\Box \Omega_1] \Box ([\Box \Omega_2])) \Box ([\Box \Omega_3]),
\]

where \( \Omega_1 = \{ a_i (p_i \parallel A \Box b_j q_j) | i < n \) and \( a_i \not\in A \}, \Omega_2 = \{ b_j (\Box a_i p_i \parallel A q_j) | j < m \) and \( b_j \not\in A \} \) and \( \Omega_3 = \{ a_i (p_i \parallel A q_j) | a_i = b_j \in A, i < n \) and \( j < m \} \).

**Proof.** Set \( N \triangleq \Box a_i p_i \parallel A \Box b_j q_j \) and \( M \triangleq (([\Box \Omega_1] \Box ([\Box \Omega_2])) \Box ([\Box \Omega_3]). \) Clearly, both \( N \) and \( M \) are stable. It is sufficient to prove \( N \sqsubseteq \text{RS} M \). Put

\[
R \triangleq \{(N, M) \} \cup \Id.
\]

We intend to check that the pair \((N, M)\) satisfies (RS1-4). For (RS1,4), it is straightforward and omitted.

**RS2** Suppose \( M \in F \). Then \( t \in F \) for some \( t \in \Omega_1 \cup \Omega_2 \cup \Omega_3 \) by Lemma 3.14. We shall consider the case where \( t \in \Omega_1 \), the others may be treated similarly and omitted. In such case, we may assume that \( t \equiv a_{i_0} (p_{i_0} \parallel A \Box b_j q_j) \) with \( i_0 < n \) and \( a_{i_0} \not\in A \). So \( p_{i_0} \not\in F \) or \( \Box b_j q_j \in F \). Clearly each of them implies \( N \equiv \Box a_i p_i \parallel A \Box b_j q_j \in F \), as desired.

**RS3** Suppose \( N \overset{\tau}{\rightarrow}_F |p'| \). Then \( M \not\in F \) by \( N \not\in F \) and (RS2). Since \( N \) is stable, \( N \overset{a_{i_0}}{\rightarrow}_F p'' \overset{s}{\rightarrow}_F |p'| \) for some \( p'' \). The proof proceeds by case analysis on the last rule applied in the proof tree of \( \text{Strip}(\text{CLL}_R, \text{M}_{\text{CLL}}) \) + \( N \overset{a_{i_0}}{\rightarrow} p'' \).

**Case 1.** \( \Box a_i p_i \parallel A \Box j < m \Box b_j q_j \overset{\tau}{\rightarrow} \) with \( \Box b_j q_j \overset{\tau}{\rightarrow} \) and \( a \not\in A \).

Then \( \Box a_i p_i \overset{a_i}{\rightarrow}_F r \) and \( p'' \equiv r \parallel A \Box b_j q_j \). By Lemma 3.14(3), we have \( a = a_{i_0} \) and \( r \equiv p_{i_0} \) for some \( i_0 < n \). Due to \( a_{i_0} = a \not\in A \), \( a_{i_0} (p_{i_0} \parallel A \Box b_j q_j) \) \( \in \Omega_1 \). So \( \Box \Omega_1 \overset{a_{i_0}}{\rightarrow} p_{i_0} \parallel A \Box b_j q_j \) by Lemma 3.14(2). Moreover, since \( \{ a_i, b_j | i < n \) and \( j < m \} \subseteq \Act \), we get \( \Box \Omega_2 \not\in F \) and \( \Box \Omega_3 \not\in F \) by Lemma 3.14(3).

Then \( M \overset{a_{i_0}}{\rightarrow} p_{i_0} \parallel A \Box b_j q_j \equiv p'' \). Hence, \( M \overset{\tau}{\rightarrow}_F |p'| \) and \( (p', p') \in R \).

**Case 2.** \( \Box j < m \Box b_j q_j \overset{\tau}{\rightarrow} \) with \( \Box a_i p_i \overset{\tau}{\rightarrow} \) and \( a \not\in A \).

Similar to Case 1.

**Case 3.** \( \Box a_i p_i \overset{a_i}{\rightarrow}_F r \Box j < m \Box b_j q_j \overset{\tau}{\rightarrow} \) with \( a \in A \).

Then \( \Box a_i p_i \overset{a_i}{\rightarrow}_F r, \Box j < m \Box b_j q_j \overset{a_i}{\rightarrow} s \) and \( p'' \equiv r \parallel A s \). By Lemma 3.14(3), we have \( a = a_{j_0}, r \equiv p_{j_0} \) for some \( i_0 < n \) and \( a = b_{j_0}, s \equiv q_{j_0} \) for some \( j_0 < m \).
Then \( a_{in}(p_{io} \parallel_A q_{jo}) \in \Omega_2 \) because of \( a_{in} = b_{jo} = a \in A \). So \( \Box \Omega_2 \xrightarrow{\omega} p_{io} \parallel_A q_{jo} \) by Lemma 3.14(2). Moreover, since \( \{a_i, b_j | i < n \text{ and } j < m \} \subseteq Act \), we get \( \Box \Omega_1 \xrightarrow{k} \Box \Omega_2 \xrightarrow{k} \) by Lemma 3.14(3). Then \( M \xrightarrow{a_{io}} p_{io} \parallel_A q_{jo} \equiv p'' \). Hence, \( M \xrightarrow{\omega} F \{ p' \} \) and \( \{ p', p'' \} \in R \).

Compared with usual expansion law in process calculus, e.g., Prop. 3.35 in [18], someone may expect that the inequation below holds, where \( \Omega_i \) \( (1 \leq i \leq 3) \) is same as ones in Prop. 3.18

\[
(\Box \Omega_1 \Box \Box \Omega_2) \Box (\Box \Omega_3) \subseteq_{RS} A \Box_{i<j} a_i, t_i \parallel_{j<m} b_j, s_j.
\]

Unfortunately, it isn’t valid. For instance, consider \( a_0, t_0 \triangleq a.\perp, a_1, t_1 \triangleq c.0 \) and \( b_0, s_0 \triangleq b.0 \) with \( a \neq b \neq c \). Let \( A = \{ a, b \} \). Clearly, the set \( \{0 \leq i \leq 3 \} \) corresponding to ones in the above proposition are: \( \Omega_1 = \{c.(0 \parallel_{(a,b)} b.0)\} \) and \( \Omega_2 = \Omega_4 = \emptyset \). Then

\[
(\Box \Omega_1 \Box \Box \Omega_2) \Box (\Box \Omega_3) \equiv (c.(0 \parallel_{(a,b)} b.0) \Box 0) \Box 0.
\]

By Lemma 3.18 \( (a.\perp \Box c.0) \parallel_{(a,b)} b.0 \in F \) and \( (c.(0 \parallel_{(a,b)} b.0) \Box 0) \Box 0 \notin F \). Then it is easy to see that \( (c.(0 \parallel_{(a,b)} b.0) \Box 0) \Box 0 \notin F \).

However, the inequation (EXP) holds for processes satisfying a moderate condition. Formally, we have the result below.

**Proposition 3.19.** Let \( n, m \geq 0, A \subseteq Act \) and \( a_i, b_j \in Act \) for each \( i < n \) and \( j < m \). Assume that \( \{ p_i \parallel_A a_i = a, b_j \in Act \) for each \( i < n \} \cup \{ q_j \parallel_{A} b_j \notin a_i \text{ for each } i < n \} \} \subseteq R \), then

\[
(\Box \Omega_1 \Box \Box \Omega_2) \Box (\Box \Omega_3) \subseteq_{RS} A \Box_{i<j} a_i, t_i \parallel_{j<m} b_j, q_j
\]

where \( \Omega_i \) \( (1 \leq i \leq 3) \) is same as ones in Prop. 3.18.

**Proof.** Set \( M \triangleq \Box_{i<n} a_i, p_i \parallel_{j<m} b_j, q_j \) and \( N \triangleq (\Box \Omega_1 \Box \Box \Omega_2) \Box (\Box \Omega_3) \). Similar to Prop. 3.18 we shall prove \( N \subseteq M \). Put \( R \triangleq \{(N, M)\} \cup Id \). It suffices to show that \( R \) is a stable ready simulation relation. We will check that the pair \( (N, M) \) satisfies (RS2), the remainder is analogous to ones of Prop. 3.18

**RS2.** Suppose \( M \in F \). By Lemmas 2.2 and 3.14 we get either \( p_{io} \in F \) for some \( i_0 < n \) or \( q_{jo} \in F \) for some \( j_0 < m \). W.l.o.g. we consider the first alternative. Then, by the assumption, \( a_{in} \notin A \) or \( a_{in} = b_{jo} \) for some \( j_0 < m \). Consequently, \( a_{in}, (p_{io} \parallel_A b_{jo}, q_j) \in \Omega_1 \) or \( a_{in}, (p_{io} \parallel_A q_{jo}) \in \Omega_3 \). Hence \( N \in F \) by Lemma 3.14 as desired.

We now have all of the properties that we need to prove the soundness of the axiomatic system \( A\text{X}_{CLL} \).

**Theorem 3.20** (Soundness). If \( \vdash p \leq q \) then \( p \subseteq_{RS} q \) for any \( p, q \in T(\Sigma_{CLL}) \).

**Proof.** Immediately follows from Lemmas 3.6.1 and 3.8. Prop. 3.7. 3.10 3.13 [15] 3.17 3.18 and 3.19 Theorem 2.7.1 and the fact that \( \subseteq_{RS} \) is reflexive and transitive.
4 Normal form and ground-completeness

This section will establish the ground-completeness of $AX_{CLL}$ for processes that are generated by BNF

$$t ::= 0 \mid \bot \mid (a.t) \mid ((\alpha.t) \mid (t \land t) \mid (t \lor t) \mid (t \parallel A)t).$$

The set of all these processes is denoted by $T(\Sigma_{CLL})$.

To prove the ground-completeness of $AX_{CLL}$, we use a standard technique involving normal forms. The idea is to isolate a particular subclass of terms, called normal forms, such that the proof of the completeness is straightforward for it. The completeness for arbitrary terms will follow if we can show that each term can be reduced to normal form using axioms and inference rules in $AX_{CLL}$. Therefore the proof of ground-completeness falls naturally into two parts: first, we will show that each process in $T(\Sigma_{CLL})$ is normalizable; second, it will be demonstrated that $AX_{CLL}$ is ground-complete w.r.t processes in normal form.

Before defining the normal form, we first introduce two useful notations.

**Notation**

1. Prefix($\square_{i<n}a_it_i) \triangleq \{a_i| i < n\}$.

2. Let $<t_0, \ldots, t_{n-1}>$ be a finite sequence of process terms with $n > 0$. The general disjunction $\bigvee_{i<n} t_i$ is defined as

$$\bigvee_{i<1} t_i \triangleq t_0, \text{ and } \bigvee_{i<k+1} t_i \triangleq (\bigvee_{i<k} t_i) \lor t_k \text{ for } k \geq 1.$$

Similar to general external choice, the order and grouping of terms in $\bigvee_{i<n}$ may be ignored by virtue of Axioms $DI1$ and $DI2$.

**Definition 4.1 (Normal Form).** The set $NF_B$ is the least subset of $T(\Sigma_{CLL})$ such that $\bigvee_{i<n} t_i \in NF_B$ if $n > 0$ and for each $i < n$, $t_i$ has the format $\square_{j<m_i} a_{ij}.t_{ij}$ with $m_i \geq 0$ such that

- (N) $t_{ij} \in NF_B$ for each $j < m_i$,
- (D) $\square_{j<m_i} a_{ij}.t_{ij}$ is injective in prefixes, and
- (N-τ) $a_{ij} \in Act$ for each $j < m_i$.

We put $NF \triangleq \{\bot\} \cup NF_B$. Each process term in $NF$ is in normal form. Notice that $NF_B \subseteq T(\Sigma_B)$, and $0 \in NF_B$ by taking $n = 1$ and $m_0 = 0$ in $\bigvee_{i<n} \square_{j<m_i} t_{ij}$.

The following simple observations inspire the format of normal processes in $NF_B$.

First, due to $\tau$-purity, the behaviour of any process consists of external and internal choices, which are interleaving but never mixing. This fact induces us to adopt the format $\bigvee_{i<n|j<m_i} \square_{t_{ij}}$ as normal forms.

Second, because of $a.p \triangleq a.(p \lor q)$ for $p, q \in T(\Sigma_B)$ and $\tau.p = RS p$, we may require normal forms to satisfy Conditions (D) and (N-τ), which make demonstrating the completeness w.r.t $NF$ (see Lemma 4.9) easier. In fact,
For each pair $j, j'$, satisfing (N-$\tau$) indeed are $\lor$-irreducible in the distributive lattice $T(\Sigma_{CLL}, \lor, \land)$ (see Remark 4.4 given later). Hence, from the lattice-theoretical viewpoint, defining normal form as above is natural.

In the following, we will show that each process term can be transformed using axioms in $AX_{CLL}$ into a normal form. To this end, the next four lemmas are firstly proved.

**Lemma 4.2.** (1) $\vdash a.t \circ a.s \leq a.(t \lor s)$.
(2) $\vdash (t \circ s_1) \lor (t \circ s_2) \leq t \circ (s_1 \lor s_2)$ for each $\circ \in \{\& \lor, \land\}$.

**Proof.** (1) $\vdash t \leq t \lor s$ and $\vdash s \leq t \lor s$ (by DI1, DI5 and TRANS).
$\Rightarrow \vdash a.t \leq a.(t \lor s)$ and $\vdash a.s \leq a.(t \lor s)$ (by CONTEXT).
$\Rightarrow \vdash a.t \circ a.s \leq a.(t \lor s)$ (by CONTEXT, EC3 and TRANS).

(2) $\vdash t \circ s_1 \leq t \circ (s_1 \lor s_2)$ and $\vdash t \circ s_2 \leq t \circ (s_1 \lor s_2)$ (by DI1, DI5 and TRANS).
$\Rightarrow \vdash t \circ s \leq t \circ (s_1 \lor s_2)$ (by CONTEXT and REF).
$\Rightarrow \vdash (t \circ s_1) \lor (t \circ s_2) \leq t \circ (s_1 \lor s_2)$ (by DI3, CONTEXT and TRANS). $\square$

The next three lemmas provide a series of closure properties of $NF$, which ensure that the inductive proof of Normal Form Theorem can be carried out smoothly.

**Lemma 4.3.** If $t, s \in NF_B$ then $\vdash t \land s = r$ for some $r \in NF$.

**Proof.** We prove it by induction on the number $|t| + |s|$. Since $t, s \in NF_B$, we may assume that $t \equiv \bigvee_{i < n} t_i$ and $s \equiv \bigvee_{i' < n'} s_{i'}$. By DI1, DI2, CO1, DS2 and Lemma 4.2(2), we get

$$\vdash t \land s = \bigvee_{i < n, i' < n'} (t_i \land s_{i'}).$$

(4.3.1)

Let $i < n$ and $i' < n'$. We will show that $\vdash t_i \land s_{i'} = r_{i,i'}$ for some $r_{i,i'} \in NF$. Clearly, we may assume that $t_i \equiv \square a_{ij} t_{ij}$ and $s_{i'} \equiv \square b_{ij} s_{ij'}$ satisfying (N), (D) and (N-$\tau$) in Def. 4.1. We consider two cases below.

Case 1. Prefix$(t_i) \neq \text{Prefix}(s_{i'}).$

By ECC1, we have $\vdash t_i \land s_{i'} = \bot$.

Case 2. Prefix$(t_i) = \text{Prefix}(s_{i'}).$

Thus, by the item (D) in Def. 4.1, we have $m_i = m_{i'}$. If $m_i = 0$ then, by the definition of general external choice, we get $t_i \equiv s_{i'} \equiv 0$. Moreover, $\vdash t_i \land s_{i'} = 0$ follows from CO2. In the following, we consider the nontrivial case where $m_i > 0$. By EC1, EC2, ECC2 and ECC3, it follows that

$$\vdash t_i \land s_{i'} = \square a_{ij}, (t_{ij} \land s_{ij'}).$$

For each pair $j, j' < m_i$ with $a_{ij} = b_{ij'}$, since $t_{ij}, s_{ij'} \in NF_B$ and $|t| + |s| > |t_{ij}| + |s_{ij'}|$, by IH, we have $\vdash t_{ij} \land s_{ij'} = t_{ij} s_{ij'}$ for some $t_{ij} s_{ij'} \in NF$. Set

$$S \equiv \square a_{ij}, t_{ij} s_{ij'}.$$
Consequently, by CONTEXT and TRANS, we have
\[ \vdash t_i \land s_{\nu} = S. \]
Clearly, if \( t_{ij'i'} \in NF_B \) for each pair \( j, j' < m \) with \( a_{ij} = b_{ij'} \), then \( S \in NF_B \). Otherwise, we have \( t_{ij'i'} \equiv \bot \) for some \( j_0, j'_0 < m \), then it follows from PR1 that
\[ \vdash a_{ij_0} \circ t_{ij'i'} \equiv \bot. \]
Further, by EC5, CONTEXT and TRANS, we get \( \vdash S = \bot \).

In summary, it follows from the discussion above that, for each \( i < n \) and \( i' < n' \),
either \( \vdash t_i \land s_{\nu} = r_{\nu} \) for some \( r_{\nu} \in NF_B \) or \( \vdash t_i \land s_{\nu} = \bot. \)
Then, by DI1, DI4 and Lemma 4.1, \( \vdash t \land s = r \) for some \( r \in NF_B \) or \( \vdash t \land s = \bot. \).

In the above proof, we do not explicitly show the proof for the induction basis where \( t \equiv s \equiv 0 \), as it is an instance of the proof of the induction step.

**Lemma 4.4.** If \( t \equiv a \circ t_i \in NF_B \) and \( s \equiv b \circ t_j \in NF_B \), then \( \vdash t \circ s = a \circ t_i \circ b \) for some \( a \circ t_i \in NF_B \).

**Proof.** If \( n = 0 \) or \( m = 0 \) then it immediately follows from EC1 and EC4 due to the definition of general external choice. In the following, we consider the non-trivial case where \( n > 0 \) and \( m > 0 \). We distinguish two cases below.

Case 1. \( \text{Prefix}(t) \cap \text{Prefix}(s) = \emptyset \).

Set
\[ p_k \triangleq \begin{cases} a_i \cdot t_k & k < n, \\ b_k \cdot s_{k-n} & n \leq k < m + n. \end{cases} \]
Then, it is trivial to check that \( \bigcirc_{k \in m+n} p_k \) satisfies (N), (D) and (N-r) in Def. 4.1, that is, \( \bigcirc_{k \in m+n} p_k \in NF_B \). Moreover, by EC2 and TRANS, it immediately follows that \( \vdash t \circ s = \bigcirc_{k \in m+n} p_k \).

Case 2. \( \text{Prefix}(t) \cap \text{Prefix}(s) \neq \emptyset \).

Let \( i_0 < n \) and \( j_0 < m \) with \( a_{i_0} = b_{j_0} \), since \( NF_B \subseteq T(\Sigma_B) \), by Lemma 4.2(1) and DS4, we get \( \vdash a_{i_0} \circ t_{i_0} \circ b_{j_0} = a_{i_0} \circ t_{i_0} \circ s_{j_0} \). Further, by Def. 4.1 DI1, DI2, CONTEXT and TRANS, it follows from \( t_{i_0}, s_{j_0} \in NF_B \) that
\[ \vdash a_{i_0} \circ t_{i_0} \circ b_{j_0} \circ s_{j_0} = a_{i_0} \circ p \text{ for some } p \in NF_B. \]

Thus, for each \( i < n \) and \( j < m \) with \( a_i = b_j \), we can fix a process term \( p_{ij} \in NF_B \) such that
\[ \vdash a_i \circ t_i \circ b_j \circ s_j = a_i \circ p_{ij}. \]

Put
\[ S_1 \triangleq a_{i} \circ \bigcirc_{i \leq n} a_t \circ s_{\nu}, \quad S_2 \triangleq b_{j} \circ \bigcirc_{j \leq m} b_s \circ p_{\mu}, \quad S_3 \triangleq a_{i} \circ \bigcirc_{a_i \in \text{Prefix}(t) \cap \text{Prefix}(s), j \leq m} a_i \circ p_{ij}. \]
Then, by EC1, EC2, TRANS and CONTEXT, we obtain $\vdash t \Box s = (S_1 \Box S_2) \Box S_3$. Clearly, both $S_1$ and $S_2$ are in $NF_B$. Moreover, since $t$ and $s$ are injective in prefixes, so is $S_3$. Hence, $S_3$ is also in $NF_B$. Further, since $\text{Prefix}(S_i) \cap \text{Prefix}(S_j) = \emptyset$ for $1 \leq i \neq j \leq 3$, similar to Case 1, we have $\vdash (S_1 \Box S_2) \Box S_3 = \Box c_i r_i$ for some $\Box c_i r_i \in NF_B$. □

**Lemma 4.5.** If $t, s \in NF_B$ then $\vdash t \parallel_A s = r$ for some $r \in NF_B$.

**Proof.** We prove it by induction on the number $|t| + |s|$. Since $t, s \in NF_B$, we may assume that $t \equiv \bigvee_{i < n} t_i$ and $s \equiv \bigvee_{i' < n'} s_i'$. By axioms DI1, DI2, PA1, DS3 and Lemma [1,2,2], we get

$$\vdash t \parallel_A s = \bigvee_{i < n, i' < n'} (t_i \parallel_A s_i').$$

(4.5.1)

We shall show that for each $i < n$ and $i' < n'$,

$$\vdash t_i \parallel_A s_i' = r_{i,i'}$$

for some $r_{i,i'} \in NF_B$.

Let $i < n$ and $i' < n'$. We may assume that $t_i \equiv \bigcirc_{j < m_i} a_{ij} t_{ij}$ and $s_i' \equiv \bigcirc_{j' < m_i'} b_{ij'} s_{ij'}$ satisfying (N), (D) and (N-τ) in Def. [4.1]. By EXP1 and EXP2, we have

$$\vdash t_i \parallel_A s_i' = (\bigcirc_{j < m_i} a_{ij} t_{ij} \parallel_A s_i') \bigcirc_{j' < m_i'} (t_i \parallel_A s_i').$$

(4.5.2)

We consider two cases.

**Case 1.** $m_i = 0$ or $m_i' = 0$.

W.l.o.g. assume that $m_i = 0$. Then, by (4.5.2), EC1, EC4, CONTEXT and TRANS, we get

$$\vdash t_i \parallel_A s_i' = (0 \parallel_A s_i').$$

(4.5.3)

If $\{b_{ij'} \notin A\} j' < m_i' \neq \emptyset$ then $\vdash t_i \parallel_A s_i' = 0$. Next, we consider the case where $\{b_{ij'} \notin A\} j' < m_i' = \emptyset$. For each $j' < m_i'$ with $b_{ij'} \notin A$, we have $s_i' = s_i'$. Moreover, $|t| + |s| > |0| + |s_i' |$. Then, by IH, we get $\vdash 0 \parallel_A s_i' = p_{i,i'}$ for some $p_{i,i'} \in NF_B$. Therefore, by CONTEXT, TRANS and (4.5.3), it is easy to see that $\vdash t_i \parallel_A s_i' = r_{i,i'}$ for some $r_{i,i'} \in NF_B$.

**Case 2.** $m_i > 0$ and $m_i' > 0$.

In such case, for each $j < m_i$ and $j' < m_i'$, we have $|t| + |s| > |t_{ij}| + |s_{ij'}|$, $|t| + |s| > |t| + |s_{ij'}|$ and $|t| + |s| > |t_{ij}| + |s_{ij'}|$. Moreover, $t_{ij}, s_{ij'}, t_{ij}, s_{ij'} \in NF_B$. Then, by IH, there exist $t_{ij'}', s_{ij'}', t_{ij'}' \in NF_B$ such that $\vdash t_{ij} \parallel_A s_i = t_{ij}'$, $\vdash t_i \parallel_A s_{ij'} = t_{ij'}'$ and $\vdash t_{ij} \parallel_A s_{ij'} = t_{ij'}'$. Set

$$S_1 \triangleq \bigcirc_{j < m_i} a_{ij} t_{ij}'', S_2 \triangleq \bigcirc_{j' < m_i'} b_{ij'}' s_{ij'}', S_3 \triangleq \bigcirc_{j' < m_i', j < m_i} a_{ij} t_{ijj'}.$$
Clearly, \( S_1, S_2, S_3 \in NF_B \) and \( \vdash t_i \parallel_A s_{i'} = (S_1 \square S_2) \square S_3 \). Further, by Lemma 4.4, we get \( \vdash t_i \parallel_A s_{i'} = r_{i''} \) for some \( r_{i''} \in NF_B \), as desired.

In summary, by the discussion above, we conclude that, for each \( i < n \) and \( i' < n', \vdash t_i \parallel_A s_{i'} = r_{i''} \) for some \( r_{i''} \in NF_B \). Then, by Def. 4.3 and (4.5.1), it immediately follows that \( \vdash t \parallel_A s = r \) for some \( r \in NF_B \), as desired. \( \blacksquare \)

Now, we can prove that each process term is normalizable. That is

**Theorem 4.6 (Normal Form Theorem).** For each \( t \in T(\Sigma_{CCL}) \), \( \vdash t = s \) for some \( s \in NF \).

**Proof.** We prove it by induction on the structure of \( t \).

- \( t \equiv 0 \) or \( t \equiv \bot \).
- Trivially.
- \( t \equiv \alpha.t_1 \).

By IH and CONTEXT, we get \( \vdash t = \alpha.t'_1 \) for some \( t'_1 \in NF \). If \( t'_1 \not\equiv \bot \) and \( \alpha \in Act \), then \( \alpha.t'_1 \in NF_B \). If \( t'_1 \equiv \bot \), by PR1, PR2 and TRANS, we obtain \( \vdash t = \bot \). If \( \alpha = \tau \), by PR2 and TRANS, we have \( \vdash t = t'_1 \).

- For \( i = 1, 2 \), by IH, we have \( \vdash t_i = t'_i \) for some \( t'_i \in NF \). We distinguish four cases based on \( \circ \).

**Case 1.** \( \circ = \lor \).  
If \( t'_1 \not\equiv \bot \) and \( t'_2 \not\equiv \bot \) (i.e., \( t'_1, t'_2 \in NF_B \)), then it immediately follows from DI1, DI2, CONTEXT and TRANS that \( \vdash t = s \) for some \( s \in NF_B \). Otherwise, w.l.o.g., assume that \( t'_1 \equiv \bot \). Then, by DI1, DI4 and TRANS, we get \( \vdash t = t'_2 \).

**Case 2.** \( \circ = \land \).  
If either \( t'_1 \equiv \bot \) or \( t'_2 \equiv \bot \), then it follows from EC1 and EC5 that \( \vdash t = \bot \). In the following, we consider the case where \( t'_1 \not\equiv \bot \) and \( t'_2 \not\equiv \bot \). In this situation, we get \( t'_1, t'_2 \in NF_B \). So, we may assume that \( t'_1 \equiv \lor \square a_{ij}.s_{ij} \) and \( t'_2 \equiv \lor \square b_{i'j'}.r_{i'j'} \) with \( \square a_{ij}.s_{ij} \) or \( \square b_{i'j'}.r_{i'j'} \in NF_B \) for each \( i < n \) and \( i' < n' \). Thus, by DI1, DI2, CONTEXT, TRANS, DS1 and Lemma 4.2(2), we obtain

\[
\vdash t_1 \sqcup t_2 = \lor i<n,i'<n' ( \square a_{ij}.s_{ij} \lor \square b_{i'j'}.r_{i'j'} ).
\]

Further, by CONTEXT, Lemma 4.4 and Def. 4.1 it immediately follows that \( \vdash t_1 \sqcup t_2 = t_3 \) for some \( t_3 \in NF_B \).

**Case 3.** \( \circ = \parallel_A \).  
If \( t'_i \in NF_B \) for \( i = 1, 2 \) then, by Lemma 4.3 we have \( \vdash t = t_3 \) for some \( t_3 \in NF \), otherwise, by CO1 and CO3, we get \( \vdash t = \bot \).

**Case 4.** \( \circ = \parallel_A \).  
If either \( t'_1 \equiv \bot \) or \( t'_2 \equiv \bot \) then, by PA1 and PA2, we get \( \vdash t = \bot \). Otherwise, we have \( t'_1, t'_2 \in NF_B \), so, by Lemma 4.5 we obtain \( \vdash t = s \) for some \( s \in NF_B \). \( \blacksquare \)
**Remark 4.7.** Clearly, \( \Box a_i.t_i =_{RS} p \lor q \) with \( a_i \in \text{Act} \) implies \( \Box a_i.t_i =_{RS} p \lor q \) or \( \Box a_i.t_i =_{RS} q \) and \( \perp =_{RS} p \lor q \) implies \( \perp =_{RS} p \) and \( \perp =_{RS} q \) for any \( p, q \).

Thus \( \perp \) and processes with form \( \Box a_i.t_i \) are \( \lor \)-irreducible in the distributive lattice \( < T(\Sigma_{CLL}), \lor, \land > \). Therefore, by the well-known result so-called Unique Decomposition Theorem in Lattice Theory (see, e.g. [3]), the normal form representation of any \( t \in T(\Sigma_{CLL}) \) is unique in an obvious sense.

We now turn our attention to the ground-completeness of \( AX_{CLL} \). First, we state a trivial result about general disjunction.

**Lemma 4.8.** Let \( n > 0 \) and \( t_i \) be stable for each \( i < n \).

1. If \( \bigvee_{i<n} t_i \notin F \) then \( \bigvee_{i<n} t_i \models_F |t_i| \) for each \( i < n \).
2. If \( \bigvee_{i<n} t_i \models |t'| \) then \( |t'| =_{t_{i_0}} \) for some \( i_0 < n \).

**Proof.** Straightforward by induction on \( n \).

A crucial step in proving the ground-completeness is to verify the completeness of \( AX_{CLL} \) w.r.t \( NF \). Next we do this.

**Lemma 4.9.** If \( t_1, t_2 \in NF \) and \( t_1 \nleq_{RS} t_2 \) then \( \vdash t_1 \leq t_2 \)

**Proof.** We prove the statement by induction on \( |t_1| \). Since \( t_1 \nleq_{RS} t_2 \), both \( t_1 \) and \( t_2 \) are stable. Further, since \( t_1, t_2 \in NF \), we get, for \( i = 1, 2 \)

\[ t_i \equiv 0 \text{ or } t_i \equiv \perp \text{ or } t_i \equiv \bigwedge_{j<n} a_{ij}.t_{ij} \in NF_B \text{ with } n_i > 0. \]  \((4.9.1)\)

Therefore, the argument splits into three cases below.

**Case 1.** \( t_1 \equiv \perp \).

Then, by \( DI1, DI4, DI5 \) and \( TRANS \), we have \( \vdash t_1 \leq t_2 \).

**Case 2.** \( t_1 \equiv 0 \).

Clearly, \( t_1 \notin F \) and \( I(t_1) = \emptyset \). Further we get \( t_2 \notin F \) and \( I(t_1) = I(t_2) \) by \( t_1 \nleq_{RS} t_2 \). Thus, by \((4.9.1)\), we have \( t_2 \equiv 0 \). Then \( \vdash t_1 \leq t_2 \) follows from \( REF \).

**Case 3.** \( t_1 \equiv \bigwedge_{i<n} a_i.t_i \) with \( n > 0 \).

Since \( t_1 \in NF_B \subseteq T(\Sigma_B) \), by Lemma \( 4.3 \) we have \( t_1 \notin F \). Hence, by \( t_1 \nleq_{RS} t_2 \), we get \( t_2 \notin F \) and \( I(t_1) = I(t_2) = \{ a_i | i < n \} \neq \emptyset \). Further, it follows from \((4.9.1)\) and the condition \( (D) \) in Def. \( 4.4 \) that there exist \( t_{2i} \in NF_B \) and \( a'_i \in Act(i < n) \) such that

\[ t_2 \equiv \bigwedge_{i<n} a'_i.t_{2i} \in NF_B \text{ and } \{ a_i | i < n \} = \{ a'_i | i < n \}. \]

By \( CONTEXT \), it is easy to know that, in order to complete the proof, it is sufficient to show that

\[ \forall i < n \exists i' < n(\vdash a_i.t_{i_0} \leq a'_{i'}.t_{2i'}). \]

Let \( i_0 < n \). We have \( a_i = a'_{i_0} \) for some \( i_0' < n \). Since \( t_{1i_0}, t_{2i_0} \in NF_B \), by Def. \( 4.1 \) there exist \( m, m' > 0, s_i(j < m) \) and \( s'_{i'}(j' < m') \) such that
1. \( t_{1i_0} \equiv \bigvee_{j<m} s_j \) and \( t_{2i_0} \equiv \bigvee_{j'<m'} s'_{j'} \).

2. \( s_j \) and \( s'_{j'} \) are stable for each \( j < m \) and \( j' < m' \).

3. \( s_j, s'_{j'} \in \text{NF}_B \) for each \( j < m \) and \( j' < m' \).

In the following, we want to show that \( \vdash s_j \leq t_{2i_0} \) for each \( j < m \). Let \( j_0 < m \). Since \( \text{NF}_B \subseteq T(\Sigma_B) \), by Lemma 6.3 and 6.8, it immediately follows that \( t_{1i_0} \supseteq_F |s_{j_0} \). Thus, \( t_1 \overset{a_{j_0}}{\rightarrow}_F t_{1i_0} \supseteq_F |s_{j_0} \). Then, it follows from \( t_1 \supseteq \sim_{RS} t_2 \) that

\[
t_2 \overset{a_{j_0}}{\rightarrow}_F |t'_2 \text{ and } s_{j_0} \supseteq \sim_{RS} t'_2 \text{ for some } t'_2.
\]

(F.9.2)

Further, since \( t_2 \) is injective in prefixes and \( t_2 \) is stable, we get \( t_2 \overset{a_{j_0}}{\rightarrow}_F t_{2i_0} \supseteq_F |t'_2 \). Then, by Lemma 4.8(2), we obtain

\[
t'_2 \equiv s'_{j'_0} \text{ for some } j'_0 < m'.
\]

(F.9.3)

Since \( |t_1| > |s_{j_0}| \), by (F.9.2), (F.9.3) and IH, we get \( \vdash s_{j_0} \leq s'_{j'_0} \). Further, by \( DI1, DI2, DI5 \) and \( \text{TRANS} \), we have \( \vdash s_{j_0} \leq t_{2i_0} \), as desired.

So far, we have obtained

\[ \vdash s_j \leq t_{2i_0} \text{ for each } j < m. \]

Then, by \( DI1, DI2, DI3, \text{CONTEXT and TRANS} \), we get \( \vdash \bigvee_{j<m} s_j \leq t_{2i_0} \), that is, \( \vdash t_{1i_0} \leq t_{2i_0} \). So, by CONTEXT, it follows that \( \vdash a_{i_0}, t_{1i_0} \leq a'_{i_0}, t_{2i_0} \).

We are now ready to prove the main result of this section.

**Theorem 4.10** (Ground-Completeness). For any \( t_1, t_2 \in T(\Sigma_{\text{CLL}}) \), \( t_1 \subseteq_{RS} t_2 \) implies \( \vdash t_1 \leq t_2 \).

**Proof.** Assume that \( t_1 \subseteq_{RS} t_2 \). By Theorem 1.6, \( \vdash t_1 = t'_1 \) and \( \vdash t_2 = t'_2 \) for some \( t'_1, t'_2 \in \text{NF} \). It suffices to prove that \( \vdash t'_1 \leq t'_2 \). By Theorem 3.20, we have \( t_1 = t'_1 \) and \( t_2 = t'_2 \). So \( t'_1 \subseteq_{RS} t'_2 \).

If \( t'_1 \equiv \perp \) then it follows from \( DI1, DI4, DI5 \) and \( \text{TRANS} \) that \( \vdash t'_1 \leq t'_2 \). Next, we consider the case \( t'_1 \not\equiv \perp \). Then, \( t'_1 \in \text{NF}_B \). We may assume \( t'_1 \equiv \bigvee_{i<n} t_{1i} \) with \( n > 0 \) and for each \( i < n, t_{1i} \equiv \bigwedge_{j<m_i} a_{ij}, r_{ij} \in \text{NF}_B \) with \( m_i \geq 0 \). In order to complete the proof, it is sufficient to show that

\[ \vdash t_{1i} \leq t_{2i} \text{ for each } i < n. \]

Let \( i_0 < n \). Since \( \text{NF}_B \subseteq T(\Sigma_B) \), by Lemma 6.3 and 6.1, we have \( t'_{1i_0} \supseteq_F |t_{1i_0} \). Then, it follows from \( t'_1 \subseteq_{RS} t'_2 \) that \( t'_2 \supseteq_F |t'_2 \) and \( t_{1i_0} \subseteq \sim_{RS} t'_2 \) for some \( t'_2 \). So, \( t'_2 \not\in F \), that is, \( t'_2 \not\equiv \perp \). Thus, \( t'_2 \in \text{NF}_B \) and we may assume that \( t'_2 \equiv \bigvee_{i<k} t_{2i} \) with \( k > 0 \) and for each \( i < k, t_{2i} \equiv \bigwedge_{j<m'_i} b_{ij}, s_{ij} \in \text{NF}_B \) for some \( m'_i \geq 0 \). Thus, \( t_{2i} \) is stable for each \( i < k \). Then, by Lemma 4.8(2), it follows from \( t'_2 \overset{a_{i_0}}{\rightarrow}_F |t'_2 \) that \( t'_{2i} \equiv t_{2i} \) for some \( i_0 < k \). Further, by Lemma 4.8(2), \( \vdash t_{1i_0} \leq t_{2i_0} \) follows from \( t_{1i_0} \subseteq \sim_{RS} t'_{2i} \). Finally, by \( DI1, DI2, DI5 \) and \( \text{TRANS} \), we obtain \( \vdash t_{1i_0} \leq t_{2i} \), as desired. \( \square \)
5 Conclusions and Discussion

This paper has provided a ground-complete proof system for weak ready simulation presented by Lüttgen and Vogler for the finite fragment of the calculus CLLR. In addition to standard axioms, since enriching process languages with logical operators conjunction and disjunction, such proof system contains a number of axioms to capture the interaction between usual process operators and logical operators.

Compared with usual notions of behaviour preorders [9], a specific point of Lüttgen and Vogler’s ready simulation is that it involves consideration of inconsistencies. The predicate \( F \) plays a central role in this notion. Due to such particular characteristic, side-conditions are attached to some axioms in AX\(_{CLL} \) (including DS\(_4\), ECC\(_3\) and EXP\(_2\)) so that processes can be treated differently according to their consistency. The guideline in designing of AX\(_{CLL} \) is that we need to find enough axioms to reduce (in)consistent processes to basic processes (\( \bot \), resp.). Such trick seems to be also useful in considering proof system for more general cases involving recursions. However, it is far from trivial to carry out this trick in the presence of recursions. In the following, we would like to discuss this sketchily.

In the framework of LLTS, since divergence is viewed as catastrophic, any process, which cannot evolve into a stable state in finitely many steps, is specified to be inconsistent. This intuition is captured formally by the condition (LTS2) in Def. 2.1. Obviously, it is recursion that may bring divergence. Thus we must put attention to such additional origin of inconsistency in the presence of recursions.

In order to carry out the trick mentioned above, we need to isolate a particular subclass of terms syntactically, which plays a role analogous to that played by \( T(\Sigma_B) \) (see Def. 3.1) in this paper. In our mind, a rational choice for such subclass is ET\(_{\Sigma_B} \) mentioned in Remark 3.2, which extends \( T(\Sigma_B) \) by admitting strongly guarded processes \( \langle X|E \rangle \) (without involving conjunction and \( \bot \)) into BNF grammar of \( T(\Sigma_B) \), and satisfies \( ET(\Sigma_B) \cap F = \emptyset \) (its proof is given in the Appendix).

To confirm that the choice above is right, we must ensure that \( ET(\Sigma_B) \) is sufficiently expressive to “represent” all consistent processes. That is, we need to provide a group of axioms so that, for any process \( t \), if \( t \) is (in)consistent then it can be reduced to one in \( ET(\Sigma_B) \) (\( \bot \) resp.) by applying these axioms. At present, it seems to be difficult to find these axioms. For instance, since there exist weakly guarded recursions that is consistent (e.g., \( \langle X|X = (X \text{div} a.0) \lor b.0 \rangle \)), we need enough axioms to transfer them into \( ET(\Sigma_B) \). In particular, a few axioms are needed to transfer (consistent) weakly guarded recursions into strongly guarded ones (notice that all recursive processes in \( ET(\Sigma_B) \) are strongly guarded). In [19], Milner has solved analogous problem for observational congruence in the calculus CCS through referring the following axioms:

\[
\begin{align*}
\langle X|X = X \square t \rangle &= \langle X|X = t \rangle & \text{(M1)} \\
\langle X|X = \tau.X \square t \rangle &= \langle X|X = \tau.t \rangle & \text{(M2)}
\end{align*}
\]

\( ^2 \)In [19], Milner uses the operator \( \rightarrow \) and the notation \( \mu X t \) instead of external choice \( \square \) and \( \langle X|X = t \rangle \) resp. Moreover Baeten and Bravetti point out that Axioms (M2) and (M3) can be equivalently expressed by a single axiom [2].
\[(X|X = \tau.(X \Box t \Box s)) = (X|X = \tau.X \Box t \Box s)\]  \hspace{1cm} (M3)

Unfortunately, none of these axioms work well in our situation. First, since unguarded recursions are incompatible with negative rules \([1]\), the calculus \(\text{CLL}_R\) restricts itself to guarded ones \([22]\). Hence Axiom (M1) is outside our terms of reference. Second, Axiom (M2) is not valid w.r.t \(=_{RS}\). For instance, consider \(t \equiv a.X\), then we get \(\langle X|X = \tau.X \Box a.X \rangle \in F\) and \(\langle X|X = \tau.a.X \rangle \notin F\). Finally, due to \(\tau\)-purity, both \(\langle X|X = \tau.(X \Box t \Box s) \rangle\) and \(\langle X|X = \tau.X \Box t \Box s \rangle\) are inconsistent for any \(t, s\). Therefore, Axiom (M3) may be useful for transferring inconsistent processes into \(\perp\) because the scope of the prefix \(\tau(.)\) in left-hand side of (M3) is larger than one in right-hand side, but it no longer has any effect on transferring consistent weakly guarded \(\langle X|E \rangle\) into strongly guarded one.

Summarily, we need to find appropriate axioms from scratch to cope with inconsistency caused by recursions.

References

[1] L. Aceto, D. de F. Escrig, C. Gregorio-Rodriguez, A. Ingolfsdottir, Axiomatizing weak simulation semantics over BCCSP, Theoretical Computer Science 537 (2014) 42-71.

[2] J.C.M. Baeten, M. Bravetti, A ground-complete axiomatisation of finite-state processes in a generic process algebra, Mathematical Structures in Computer Science 18 (2008) 1057-1089.

[3] G. Birkhoff, Lattice Theory, American Mathematical Society, 1948.

[4] B. Bloom, Ready simulation, bisimulation, and the semantics of the CCS-like languages, Ph.D dissertation, MIT, 1994.

[5] B. Bloom, S. Istrail, A. Meyer, Bisimulation can’t be traced, Journal of the ACM 42 (1995) 232-268.

[6] R. Bol, J.F. Groote, The meaning of negative premises in transition system specifications, Journal of the ACM 43 (1996) 863-914.

[7] R. Cleaveland, G. Lüttgen, A semantic theory for heterogeneous system design, in: FSTTCS 2000, in: LNCS, vol. 1974, Springer-Verlag, 2000, pp. 312-324.

[8] R. Cleaveland, G. Lüttgen, A logical process calculus, in: EXPRESS 2002, in: ENTCS, 68, 2, Elsevier Science, 2002.

[9] R.J. van Glabbeek, The linear time - branching time spectrum I, in: J.A. Bergstra, A. Ponse, S.A. Smolka, (Eds.), Handbook of Process Algebra, Chapter 1, Elsevier Science, 2001, pp. 3-100.

[10] S. Graf, J. Sifakis, A logic for the description of non-deterministic programs and their properties, Information Control 68 (1986) 254-270.

[11] C.A.R. Hoare, Communicating sequential processes, Prentice-Hall, 1985.

[12] R. Kurshan, Computer-Aided Verification of Coordinating Processes: The Automata-Theoretic Approach, Princeton Univ. Press, 1994.
[13] K.G. Larsen, A. Skou, Bisimulation through probabilistic testing, Information and computation 94 1-28.

[14] H.M. Lin, Complete inference systems for weak bisimulation equivalences in the π-calculus, Information and computation 180(1) (2003) 1-29.

[15] G. Lüttgen, W. Vogler, Conjunction on processes: full-abstraction via ready-tree semantics, Theoretical Computer Science 373 (1-2) (2007) 19-40.

[16] G. Lüttgen, W. Vogler, Ready simulation for concurrency: it’s logical, Information and computation 208 (2010) 845-867.

[17] G. Lüttgen, W. Vogler, Safe reasoning with Logic LTS, Theoretical Computer Science 412 (2011) 3337-3357.

[18] R. Milner, Communication and Concurrency, Prentice Hall, 1989.

[19] R. Milner, A complete axiomatization for observational congruence of finite-state behaviours, Information and Computation 81 (1989) 227-247.

[20] E. Olderog, Nets, Terms and Formulas in: Cambridge Tracts in Theoretical Computer Science, vol. 23, Cambridge Univ. Press, 1991.

[21] D.A. Peled, Software Reliability Methods, Springer-Verlag, 2001.

[22] Y. Zhang, Z.H. Zhu, J.J. Zhang, On recursive operations over Logic LTS, Mathematical Structures in Computer Science (in press, doi:10.1017/S0960129514000073), also see arXiv:1301.3350.

A Appendix

We mentioned in Section 5 that $ET(\Sigma_B) \cap F = \emptyset$. This Appendix is devoted to proving this claim. We first define $ET(\Sigma_B)$ formally.

Definition A.1 (Extended Basic Term). The extended basic terms are defined by BNF: $t ::= 0 \mid (\alpha.t) \mid t\&t \mid t \lor t \mid t \parallel X \mid (X|X = t)$, where $\alpha \in \text{Act}$, $X \in \text{VAR}$, $A \subseteq \text{Act}$ and in $(X|X = t)$, $X$ is strongly guarded in $t$. We denote $ET(\Sigma_B)$ as the set of all extended basic terms.

As usual, we use $t_X$ to denote a term $t$ whose free variables form a subset of $\{X_1, \ldots, X_n\}$ where $X = (X_1, \ldots, X_n)$ is a $n$-tuple distinct variables. $t_X$ is stable if $t_X\{\tilde{\tau}\}/\tilde{X}$.

Lemma A.2. If $t_X$ is stable then $t_X\{\tilde{p}/\tilde{X}\} \not\rightarrow r$ for any $\tilde{p}$.

Proof. Assume $t_X\{\tilde{p}/\tilde{X}\} \not\rightarrow r$ for some $r$. It suffices to prove $t_X\{\tilde{\tau}/\tilde{X}\} \not\rightarrow$. It proceeds by induction on the depth of the inference of $Strip(\text{CLL}_R, M_{\text{CLL}_R}) \vdash t_X\{\tilde{p}/\tilde{X}\} \rightarrow r$. The induction is easy to carry out by distinguishing several cases based on the last rule applied in the inference. We leave the proof to the reader. □
Lemma A.3. If $t_X$ is a term in $ET(\Sigma_B)$ such that $X$ is strongly guarded in $t_X$ for each $X \in \vec{X}$, then there exists $t'_X \in ET(\Sigma_B)$ such that, for any $\vec{q}$, $t_X(\vec{q}/\vec{X}) \Rightarrow |t'_X(\vec{q}/\vec{X})|$. 

Proof. If $t_X$ is stable then the conclusion holds trivially by Lemma A.2. In the following, we devote ourselves to considering non-trivial case where $t_X$ is not stable. It proceeds by induction on the structure of $t_X$. Here we consider only non-trivial case $t_X \equiv \langle Y|Y = t \rangle$. In this situation, $Y \notin \vec{X}$ and $t$ is in $ET(\Sigma_B)$ whose free variables are in $\{Y\}$ $\cup \vec{X}$. Moreover, for each $Z \in \{Y\} \cup \vec{X}$, $Z$ is strongly guarded in $t$. Hence, by IH, there exists $t' \in ET(\Sigma_B)$ such that $t\{\vec{p}/\vec{X}, q/Y\} \Rightarrow |t'\{\vec{p}/\vec{X}, q/Y\}|$ for any $\vec{p}, q$. In particular, we get

$$t\{\vec{p}/\vec{X}, \langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\} \Rightarrow |t'\{\vec{p}/\vec{X}, \langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\}|$$

Further, by Rule Ra16, it follows from $t\{\vec{p}/\vec{X}, \langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\} \equiv t\{\langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\}\}$ that

$$\langle Y|Y = t \rangle\{\vec{p}/\vec{X}\} \Rightarrow |t'\{\langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\}|$$

Set $t''_X \doteq t'\{\langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\}|$. Then it is easy to see that $t''_X \in ET(\Sigma_B)$ due to $t'\{\langle Y|Y = t \rangle\{\vec{p}/\vec{X}/Y\}| \in ET(\Sigma_B)$. Hence $t''_X$ is the one that we desire. 

As an immediate consequence of the lemma above, we have

Corollary A.4. For any process (i.e., terms with no free variables) $p \in ET(\Sigma_B)$, there exists $q \in ET(\Sigma_B)$ such that $p \Rightarrow |q|$. 

Proposition A.5. $ET(\Sigma_B) \cap F = \emptyset$. 

Proof. Since $F$ is a set of processes, it suffices to show that each process in $ET(\Sigma_B)$ is consistent. Let $\Omega$ be the set of all processes in $ET(\Sigma_B)$. Due to the well-foundedness of proof trees, in order to complete the proof, it is sufficient to show that, for any $p \in \Omega$, if $\mathcal{T}$ is a proof tree of $\text{Strip}(\text{CLL}_R, M_{\text{CLL}_A}) \vdash pF$ then $\mathcal{T}$ has a proper subtree with root $rF$ for some $r \in \Omega$. We shall prove this as follows.

Let $p \in \Omega$ and $\mathcal{T}$ be a proof tree of $pF$. It is a routine case analysis based on the last rule applied in $\mathcal{T}$. We distinguish different cases based on the form of $p$. Clearly, $p \neq 0$ due to $0 \notin F$. For $p \equiv \alpha.p_1$ or $p_1 \circ p_2$ with $\circ \in \{\lor, \land\}$, it is obvious that $p_1, p_2 \in ET(\Sigma_B)$. Moreover, by SOS rules of CLL$_R$, it is easy to see that $\mathcal{T}$ has a proper subtree with root $p_iF$ for some $i \in \{1, 2\}$. Next we handle the case $p \equiv \langle Y|Y = t_Y \rangle$. Then the last rule applied in $\mathcal{T}$ is either $\langle Y|Y = t_Y \rangle \Rightarrow |rF|Y|Y = t_Y|F \Rightarrow rF|Y|Y = t_Y|F$ or $\Rightarrow |rF|Y|Y = t_Y|F \Rightarrow rF|Y|Y = t_Y|F$. For the former, it is obvious that $\langle Y|Y = t_Y \rangle \in \Omega$ due to $\langle Y|Y = t_Y \rangle \equiv t_Y\{\langle Y|Y = t_Y \rangle\{Y|Y = t_Y\}/Y\}$ (see subsection 2.2) and $t_Y, \langle Y|Y = t_Y \rangle \in ET(\Sigma_B)$. For the latter, by Corollary A.4, $\langle Y|Y = t_Y \rangle \Rightarrow |r'|$ for some $r' \in \Omega$, as desired. 

25