特集●ソフトウェア工学の基礎

A Non-interleaving Timed Process Algebra and a Process Logic for Verifying Composition of Agents

Yoshinao Isobe Kazuhito Ohmaki

We present formal frameworks *tCCA*, *tLCA*, and *tICCA* for verifying composition of agents. Behaviors of composite agents are described in *tCCA* and specifications for them are described in *tLCA*. Since consistency between specifications in *tLCA* is undecidable as proven in this paper, we propose to use intermediate specifications described in *tICCA* instead of directly checking the consistency, and then give useful propositions for verifying composition of agents in *tICCA*.

1 Introduction

We are members of a project called ESP (Evolutionary System Project). The purpose of ESP is to develop an agent-system where agents can evolve by spontaneously moving over networks, combining with other agents, and communicating with them. In such systems, unexpected behavior such as deadlocks may be caused by composition of agents. To avoid unexpected behavior, the members of ESP are discussing an agent-system where each agent has specifications, and if specifications between two agents are not consistent, then they cannot combine. These agents are modeled as shown in Fig.1. In ESP, we are studying how to formally verify consistency between specifications. This paper presents a process algebra *tCCA* to describe behaviors of composite agents, a process logic *tLCA* to describe specifications, and a process algebra *tICCA* to describe intermediate specifications, and then shows useful propositions for effectively verifying composition of agents.

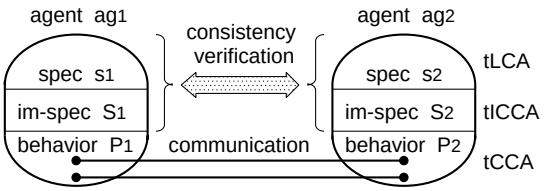

Fig.1 The image of agents with specifications

tCCA (a timed Calculus of Composite Agents) is a non-interleaving timed process algebra. Many non-interleaving process algebras have already been proposed (e.g. in [1][4][8][14]) by considering locality or causality between actions, and many timed process algebras also have been proposed (e.g. in [3][5][15][20]). The features of *tCCA* are summarized as follows:

(1) **non-interleaving:** it is possible to observe which agents have performed actions,

(2) **durational action:** the passage of time is needed for executing any action.

By the feature (1), concurrent behavior is explic-

エージェントの合成を検証するための非インターリービング時間付プロセス代数とプロセス論理

磯部 祥尚, 大蒔 和仁, 産業技術総合研究所, National Institute of Advanced Industrial Science and Technology.

コンピュータソフトウェア, Vol.20, No.3 (2003), pp.0–0.
2003 年 1 月 10 日受付.

itly distinguished from interleaving behavior. The feature (2) removes unnatural situations such that actions can infinitely be performed in a finite time. Note that each feature is not new. For example, (1) is expressed in Distributed CCS [8] and (2) is discussed in [3]. The purpose of this paper is not to propose a new process algebra, but to show relations between a process algebra and a process logic with considering both of the features (1) and (2).

tLCA (a timed Logic for Composite Agents) is a process logic for describing specifications for agent-behaviors described in *tCCA*. As shown in Fig.1, if each agent ag_i has a specification s_i (i.e. each behavior P_i satisfies s_i), then it is important to verify the consistency between s_1 and s_2 before composition of ag_1 and ag_2. However, we prove that the consistency is undecidable in Section 4.

tICCA (a timed Intermediate Calculus of Composite Agents) is a timed basic process algebra. Since the consistency in *tLCA* is undecidable, we use *intermediate specifications* (also called *im-specifications*) between behaviors (*tCCA*) and specifications (*tLCA*). Im-specifications express *abstract* behaviors of agents by hiding uninteresting actions because verification often considers some actions (not all actions). Therefore, the number of states in im-specifications can decrease. In Section 5, we show how to verify composition of agents in *tICCA* instead of in *tCCA*.

The outline of this paper is as follows: Sections 2 and 3 define *tCCA* and *tLCA*, respectively. Section 4 shows that satisfiability in *tLCA* is undecidable using undecidability of the membership problem in unrestricted grammars [9]. Then, Section 5 presents *tICCA* to use im-specifications. Section 6 and Section 7 discuss related works and our approach, respectively. Section 8 concludes this paper. The omitted proofs are given in the Appendix.

2 Process Algebra *tCCA*

In this section, *tCCA* is defined for describing behaviors of composite agents. At first, it is assumed that a set $\mathcal{N}_{ac}$, ranged over by $a, b, \cdots$, of *action-names* are given. Then, the set Act, ranged over by $\alpha, \beta, \cdots$, of *actions* is defined as: $Act = \mathcal{N}_{ac} \cup \{\mathtt{1}\}$, where the *time action* '$\mathtt{1}$' represents the passage of *one time-unit*, where a time-unit is an abstract unit of time, and it may be a second, a minute, or a clock of CPU. Also, it is assumed that a set $\mathcal{N}_{ag}$, ranged over by $\psi, \varphi, \cdots$, of *agent-names* are given. Then, the set Act_S of *single-actions* is defined as: $Act_S = \{a@\psi : a \in \mathcal{N}_{ac}, \psi \in \mathcal{N}_{ag}\}$, and subsets of Act_S are represented by $\mu, \nu, \cdots$. Especially, if a subset μ of Act_S contains one single-action $a@\psi$, at most, for each agent-ψ, then μ is a *multi-action*. Thus, the set Act_M of multi-actions is defined as:

$$Act_M = \{\mu : \mu \subseteq Act_S, \forall a@\psi \in \mu. \\ \forall b@\varphi \in \mu - \{a@\psi\}.\ \psi \neq \varphi\}$$

A multi-action $\{a_i@\psi_i : i \in I\} \in Act_M$ represents that agents-ψ_i ($i \in I$) simultaneously perform actions-a_i, respectively, where *action-a* and *agent-ψ* are an action named a and an agent named ψ, respectively. Thus, the number of actions which an agent can simultaneously perform is one at most. Subsets of the set Act_M of multi-actions are represented by $M, N, \cdots$.

It is also assumed that a set $\mathcal{K}_{bh}$, ranged over by $K, \cdots$, of *behavior-constants* is given. Then, the set Bh, ranged over by $P, Q, \cdots$, of *behaviors* is the smallest set which contains the following expressions:

K : Recursion ($K \in \mathcal{K}_{bh}$),
$\mathbf{I}$: Idling,
$a.P$: Prefix ($a \in \mathcal{N}_{ac}$),
$P \triangleright Q$: Timeout,
$\sum_{i \in I} P_i$: Summation (I an indexing set),

where P, P_i, Q are already in Bh. Also, the set CBh, ranged over by $C, D, \cdots$, of *concurrent behav-*

iors is the smallest set which contains the following expressions:

$P@\psi$: Naming $(P \in Bh, \psi \in \mathcal{N}_{ag})$,

$C|D$: Composition $(Agn(C) \cap Agn(D) = \emptyset)$,

$C \backslash M$: Restriction $(M \subseteq Act_M)$,

where C, D are already in CBh. The function $Agn : CBh \rightarrow 2^{\mathcal{N}_{ag}}$ is defined as follows: $Agn(P@\psi) = \{\psi\}$, $Agn(C|D) = Agn(C) \cup Agn(D)$, and $Agn(C \backslash M) = Agn(C)$. To avoid too many parentheses, these operators have the binding power such that: Prefix > Timeout > Summation > Naming > Restriction > Composition. Notation $C \equiv D$ represents that C and D are syntactically identical.

Each operator is briefly explained as follows. The behavior $a.P$ represents that an agent can perform action-a and thereafter behaves like P. Here, it is important that the passage of one time-unit is always necessary for executing any action. An action $a(n)$ which needs n time-units $(n \geq 1)$ for its execution is inductively defined as follows:

$$a(1).P \equiv a.P,$$
$$a(n+1).P \equiv a.a(n).P.$$

Here, note that 'a' of $a.P$ does not represent '$\mathtt{1}$'. Alteration by time is represented by the Timeout operator $\triangleright$.

$\sum_{i \in I} P_i$ behaves like P_j for some $j \in I$, and the choice is made by the first action of P_j except for the time action $\mathtt{1}$. The passage of time must be made by all behaviors P_i $(i \in I)$. This operator corresponds to 'strong choice' in TCCS [15]. We also use a short notation for binary choice as follows: $P_1 + P_2 \equiv \sum_{i \in \{1,2\}} P_i$.

Meaning of each behavior-constant is given by a defining equation. It is assumed that for every behavior-constant $K \in \mathcal{K}_{bh}$, there is a defining equation of the form $K \stackrel{\text{def}}{=} P$ for some $P \in Bh$, where P can contain behavior-constants. Thus, we can express recursive behavior by behavior-constants.

$P \triangleright Q$ *now* behaves like P and behaves like Q *after one time-unit*. Thus, $P \triangleright_{(n)} Q$ which behaves like P until n time-units pass and behaves like Q after that is defined as:

$$P \triangleright_{(0)} Q \equiv Q,$$
$$P \triangleright_{(n+1)} Q \equiv P \triangleright (P \triangleright_{(n)} Q).$$

Furthermore, $(n).P$ which behaves like P after n time-units are defined as: $(n).P \equiv \mathbf{I} \triangleright_{(n)} P$.

The Naming operator @ defines that an agent named ψ behaves like P by $P@\psi$. The function Agn is used for uniquely assigning an agent-name. The concurrent behavior $(C|D)$ concurrently executes C and D. $C \backslash M$ can execute only multi-actions contained in the set $M \cup \{\emptyset\}$.

Note that Prefix is not $\mu.P$ but $a.P$. For example, $(a.b.\mathbf{I}@\psi | c.\mathbf{I}@\varphi) \setminus \{\mu_1, \mu_2\}$ cannot be expanded to $\mu_1.\mu_2.\mathbf{I}$, where $\mu_1 = \{a@\psi\}$ and $\mu_2 = \{b@\psi, c@\varphi\}$. It means that the number of actions which an agent can simultaneously perform is one at most.

Semantics of behaviors and concurrent behaviors is given by the labelled transition systems $\langle Bh, Act, \{\stackrel{\alpha}{\longrightarrow} : \alpha \in Act\}\rangle$ and $\langle CBh, Act_M, \{\stackrel{\mu}{\longrightarrow} : \mu \in Act_M\}\rangle$, respectively, where the sets $\stackrel{\alpha}{\longrightarrow}$ and $\stackrel{\mu}{\longrightarrow}$ of transitions are the smallest relations satisfying the inference rules in Figs.2 and 3.

As shown in the rules **Id**, $\mathbf{Act}_2$, $\mathbf{Sum}_2$, **Rec** in Fig.2, behavior of agents does not alter by the passage of time $\mathtt{1}$ without the Timeout operator $\triangleright$. The rules $\mathbf{TO}_{1,2}$ show that the timeout process of P is ignored in $P \triangleright Q$, and the timeout process of $P \triangleright Q$ is Q.

The rule $\mathbf{Name}_1$ in Fig.3 assigns an agent-name ψ to an action a performed by the agent-ψ, and **Com** combines all simultaneously performed actions into a multi-action. On the other hand, the passage of time cannot be observed as shown in $\mathbf{Name}_2$. Thus, the passage of one time-unit for behaviors is $\stackrel{\mathtt{1}}{\longrightarrow}$, and one for concurrent behaviors is $\stackrel{\emptyset}{\longrightarrow}$. From the inference rules in Figs.2 and 3, it is easily proven that the passage of time is always possible (i.e. for any C, for some C', $C \stackrel{\emptyset}{\longrightarrow} C'$), and alteration by the passage of time is determinate

Name	Hypothesis	⇒	Conclusion
Id		⇒	$\mathbf{I} \xrightarrow{1} \mathbf{I}$
$\mathbf{Act}_1$		⇒	$a.P \xrightarrow{a} P$
$\mathbf{Act}_2$		⇒	$a.P \xrightarrow{1} a.P$
$\mathbf{Sum}_1$	$\exists i \in I.\ P_i \xrightarrow{a} P'$	⇒	$\sum_{i\in I} P_i \xrightarrow{a} P'$
$\mathbf{Sum}_2$	$\forall i \in I.\ P_i \xrightarrow{1} P_i'$	⇒	$\sum_{i\in I} P_i \xrightarrow{1} \sum_{i\in I} P_i'$
$\mathbf{TO}_1$	$P \xrightarrow{a} P'$	⇒	$P \triangleright Q \xrightarrow{a} P'$
$\mathbf{TO}_2$		⇒	$P \triangleright Q \xrightarrow{1} Q$
Rec	$K \stackrel{\text{def}}{=} P,\ P \xrightarrow{\alpha} P'$	⇒	$K \xrightarrow{\alpha} P'$

Fig.2 Inference rules for $\xrightarrow{\alpha} \subseteq Bh \times Bh$

(note: $a \in \mathcal{N}_{ac}, \alpha \in Act = \mathcal{N}_{ac} \cup \{1\}$**)**

Name	Hypothesis	⇒	Conclusion
$\mathbf{Name}_1$	$P \xrightarrow{a} P'$	⇒	$P@\psi \xrightarrow{\{a@\psi\}} P'@\psi$
$\mathbf{Name}_2$	$P \xrightarrow{1} P'$	⇒	$P@\psi \xrightarrow{\emptyset} P'@\psi$
Com	$C \xrightarrow{\mu} C', D \xrightarrow{\nu} D'$	⇒	$C\|D \xrightarrow{\mu\cup\nu} C'\|D'$
Res	$C \xrightarrow{\mu} C', \mu \in M \cup \{\emptyset\}$	⇒	$C\backslash M \xrightarrow{\mu} C'\backslash M$

Fig.3 Inference rules for $\xrightarrow{\mu} \subseteq CBh \times CBh$

(i.e. if $C \xrightarrow{\emptyset} C'$ and $C \xrightarrow{\emptyset} C''$, then $C' \equiv C''$).

We also use a behavior-constant $P \rangle Q$ for each $P, Q \in Bh$ defined as follows:

$$P \rangle Q \stackrel{\text{def}}{=} \sum\{a.(P' \rangle Q') : P \xrightarrow{a} P', Q \xrightarrow{1} Q'\} \cup \{a.Q' : Q \xrightarrow{a} Q'\} \triangleright \sum\{(P' \rangle Q') : P \xrightarrow{1} P', Q \xrightarrow{1} Q'\}$$

The constant $(P \rangle Q)$ represents an *interrupt* for P by Q. Thus, it usually behaves like P but behaves like Q after Q has performed an action (not $\mathtt{1}$). In this definition, Q can alter $(Q \xrightarrow{1} Q')$ by the passage of time even before the interrupt.

In the rest of this section, we give an example of expression in *tCCA*. It is interaction between a researcher and a pizza-worker. In this example, a time-unit is a minute. At first, the behavior *RES* of the researcher is defined as follows:

$$RES \stackrel{\text{def}}{=} order.(\ study(30).\mathbf{I} \rangle (receive.eat(20).\mathbf{I} \triangleright_{(60)} cancel.angry(10).\mathbf{I}))$$

The researcher *orders* a pizza, and then *studies* for 30 minutes, but if he *receives* the pizza, then he stops studying and *eats* the pizza for 20 minutes. However, if 60 minutes have passed, then he *cancels* the order and is *angry* for 10 minutes. For example, the following transition is possible:

$$RES \xrightarrow{order} (\xrightarrow{study})^{15} \xrightarrow{receive} (\xrightarrow{eat})^{20} \mathbf{I},$$

where $(\xrightarrow{\alpha})^n$ represents $\xrightarrow{\alpha} \cdots \xrightarrow{\alpha}$ with n occurrences of $\xrightarrow{\alpha}$. In addition, since the researcher can rest during study, he may cancel the order before finishing the study as follows:

$$RES \xrightarrow{order} (\xrightarrow{study})^{10} (\xrightarrow{1})^{45} (\xrightarrow{study})^{5} \xrightarrow{cancel} (\xrightarrow{angry})^{10} \mathbf{I},$$

where $(\xrightarrow{1})^{45}$ means that the researcher does not anything for 45 minutes. This idle time (i.e. $study(20).\mathbf{I} \xrightarrow{1} study(20).\mathbf{I}$) is inferred by $\mathbf{Act}_2$. Note that n of the Timeout operator $\triangleright_{(n)}$ decreases without respect to his actions by the definition of the constant $(\cdots\rangle\ (\cdots \triangleright_{(n)} \cdots))$.

Next, the behavior *PIZ* of the pizza-worker is defined as follows:

$$PIZ \stackrel{\text{def}}{=} accept.(\ bake(15).drive(10).deliver.\mathbf{I} \rangle canceled.\mathbf{I})$$

If the pizza-worker *accepts* an order, then he *bakes* a pizza for 15 minutes, and then *drives* for 10 minutes, and finally *delivers* the pizza. However, the order may always be *canceled*. In the fastest case, the pizza-worker can deliver the pizza after 25 minutes ($bake(15)$ and $drive(10)$), but the order may be canceled if he is idle as follows:

$$PIZ \xrightarrow{accept} (\xrightarrow{1})^{10} (\xrightarrow{bake})^{15} (\xrightarrow{1})^{30} (\xrightarrow{drive})^{5} \xrightarrow{canceled} \mathbf{I}.$$

Finally, the researcher and the pizza-worker are concurrently combined as follows:

$$RP \equiv (RES@bob|PIZ@john)\backslash M_{rp}$$

where *bob* and *john* are names of the researcher and the pizza-worker, respectively. And M_{rp} is the set of feasible multi-actions in *RP* and is given as follows:

$$M_{rp} = M_{sync} \cup M_r \cup M_p \cup \{\mu \cup \nu : \mu \in M_r, \nu \in M_p\}$$

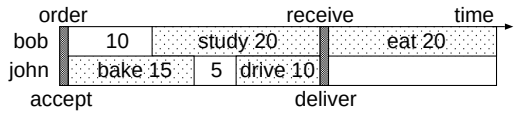

Fig.4 A transition-sequence by *RP*

where M_{sync}, M_r, and M_p are given as follows:

$$M_{sync} = \{\{order, accept\}, \{receive, deliver\}, \{cancel, canceled\}\}$$
$$M_r = \{\{study\}, \{eat\}, \{angry\}\},$$
$$M_p = \{\{bake\}, \{drive\}\}$$

where agent-names *bob* and *john* to be appended to each action are omitted (e.g. $\{order, accept\}$ is the abbreviation of $\{order@bob, accept@john\}$). M_{sync} is the set of multi-actions which must synchronize between the researcher and the pizza-worker. M_r and M_p are the sets of multi-actions which can independently or simultaneously be performed by the researcher and the pizza-worker, respectively. For example, an order must synchronize with an acceptance, and the pizza-worker can bake and drive while the researcher is studying. For example, the following transition can be inferred and it means that the researcher rests for 10 minutes after the order, and then studies for 20 minutes, and then receives the pizza, and then eats it.

$$RP \xrightarrow{\{order,accept\}} (\xrightarrow{\{bake\}})^{10} (\xrightarrow{\{study,bake\}})^{5} (\xrightarrow{\{study\}})^{5} (\xrightarrow{\{study,drive\}})^{10} \xrightarrow{\{receive,deliver\}} (\xrightarrow{\{eat\}})^{20} (\mathbf{I}@bob|\mathbf{I}@john)\backslash M_{rp}$$

This transition-sequence is depicted in Fig.4.

3 A Process Logic *tLCA*

Although specifications can be expressed in process algebras, process logics such as Hennessy-Milner logic [12] or μ-calculus [18] have an advantage of not always requiring to specify the full behavior of systems. For example, in process logics, specifications can be loosely expressed by possibilities $\langle a \rangle$ or disjunctions $\vee$; and they can be refined by necessities $[a]$ and conjunctions $\wedge$. In this section, a process logic *tLCA* is defined to describe specifications for concurrent behaviors in *tCCA*.

First, it is assumed that a set $\mathcal{K}_{sp}$, ranged over by $K, \cdots$, of *specification-constants* (also called *Constants*) is given. Then, the set *Sp*, ranged over by $s, t, \cdots$, of *specifications* is the smallest set which contains the following expressions:

$$K : \text{Recursion } (K \in \mathcal{K}_{sp}),$$
$$\langle \mu \rangle s : \text{Possibility } (\mu \in Act_M),$$
$$[\mu] s : \text{Necessity } (\mu \in Act_M),$$
$$\textstyle\bigwedge_{i \in I} s_i : \text{Conjunction } (I \text{ an indexing set}),$$
$$\textstyle\bigvee_{i \in I} s_i : \text{Disjunction } (I \text{ an indexing set}),$$

where s, s_i are already in *Sp*. And subsets of *Sp* are represented by $\sigma, \rho, \cdots$. These operators have the binding power such that: {Possibility, Necessity} > Conjunction > Disjunction. The following short notations are also used for the true $\mathtt{tt}$, the false $\mathtt{ff}$, and so on.

$$\mathtt{tt} \equiv \textstyle\bigwedge_{i \in \emptyset} s_i, \qquad s_1 \wedge s_2 \equiv \textstyle\bigwedge_{i \in \{1,2\}} s_i,$$
$$\mathtt{ff} \equiv \textstyle\bigvee_{i \in \emptyset} s_i, \qquad s_1 \vee s_2 \equiv \textstyle\bigvee_{i \in \{1,2\}} s_i,$$
$$\langle \mu \rangle^1 s \equiv \langle \mu \rangle s, \qquad \langle \mu \rangle^{n+1} s \equiv \langle \mu \rangle \langle \mu \rangle^n s,$$
$$[\mu]^1 s \equiv [\mu] s, \qquad [\mu]^{n+1} s \equiv [\mu][\mu]^n s,$$
$$\langle a_i@\psi_i : i \in I \rangle s \equiv \langle \{a_i@\psi_i : i \in I\} \rangle s,$$
$$[a_i@\psi_i : i \in I] s \equiv [\{a_i@\psi_i : i \in I\}] s.$$

The Possibility $\langle a_i@\psi_i : i \in I \rangle s$ requires that all agents-ψ_i ($i \in I$) can simultaneously perform a_i respectively, and then s can be satisfied after that. The Necessity $[a_i@\psi_i : i \in I] s$ requires that if all agents-ψ_i simultaneously perform a_i respectively, then s must always be satisfied after that. Since $C \xrightarrow{\emptyset} C'$ is always possible and determinate, the requirements of $\langle \emptyset \rangle s$ and $[\emptyset] s$ are equal.

It is assumed that for every Constant $K \in \mathcal{K}_{sp}$, there is a defining equation $K \stackrel{\text{def}}{=} s$ for some $s \in Sp$, and each Constant is guarded by Possibilities or Necessities. For example, $K \stackrel{\text{def}}{=} \langle \mu \rangle K$ is guarded and $K \stackrel{\text{def}}{=} K \wedge \langle \mu \rangle K$ is not guarded. This recursive requirement corresponds to maximum fixpoint. Since *tLCA* has no minimum fixpoint, it cannot be expressed that a specification s is *eventually* satisfied (in a finite expression), but it can be expressed that s is satisfied *by* n time-units pass instead.

Next, a function $DC : Sp \to 2^{2^{Sp}}$ is inductively defined as follows:

$$DC(\langle\mu\rangle s) = \{\{\langle\mu\rangle s\}\},$$
$$DC([\mu]s) = \{\{[\mu]s\}\},$$
$$DC(\textstyle\bigwedge_{i\in I} s_i) = \{\bigcup_{i\in I} \sigma_i : \forall i \in I.\ \sigma_i \in DC(s_i)\},$$
$$DC(\textstyle\bigvee_{i\in I} s_i) = \bigcup_{i\in I} DC(s_i),$$
$$DC(K) = DC(s), \text{ if } K \stackrel{\text{def}}{=} s.$$

This function is used for translating any specification s into a disjunctive normal form such that $\bigvee\{\bigwedge\{s' : s' \in \sigma\} : \sigma \in DC(s)\}$.

Then, satisfaction relation between a concurrent behavior (in *tCCA*) and a specification (in *tLCA*) is defined by using this function DC.

Definition 3.1 *Let $\nu \subseteq Act_S$. A set $\mathcal{R} \subseteq CBh \times Sp$ is a (ν)-satisfaction-subset, if $(C, s) \in \mathcal{R}$ implies that for some $\sigma \in DC(s)$, the following conditions (i) and (ii) hold for every $\mu, \mu'' \in Act_M$, $s' \in Sp$, and $C'' \in CBh$,*

(i) if $\langle\mu\rangle s' \in \sigma$, then for some C' and μ',
$C \xrightarrow{\mu'} C'$, $\mu \doteq_\nu \mu'$, *and* $(C', s') \in \mathcal{R}$,

(ii) if $[\mu]s' \in \sigma$, $C \xrightarrow{\mu''} C''$, and $\mu \doteq_\nu \mu''$,
then $(C'', s') \in \mathcal{R}$,

where the (ν)-restricted equivalence relation $\doteq_\nu$ over multi-actions is defined as follows:

$$\mu_1 \doteq_\nu \mu_2 \text{ if and only if } \mu_1 \cap \nu = \mu_2 \cap \nu.$$

Then, if $(C, s) \in \mathcal{R}$ for some (ν)-satisfaction-subset $\mathcal{R}$, then C satisfies *s with respect to ν, written $C \models_\nu s$. For the special case $\nu = Act_S$, $C \models_{Act_S} s$ is abbreviated to $C \models s$.* ∎

The parameter ν in $\models_\nu$ is used for *partial verification* and the set ν of single-actions are called an *available set*. Although $\models_\nu$ can be indirectly defined from $\models$ (e.g. $C \models_\nu \langle\mu\rangle\mathtt{tt}$ iff $C \models \bigvee\{\langle\mu'\rangle\mathtt{tt} : \mu \doteq_\nu \mu' \in Act_M\}$), we directly define $\models_\nu$ for convenience and effective verification. The satisfaction relation $\models_\nu$ is the largest (ν)-satisfaction-subset and such definition style is similar to one of bisimilarity [12]. This satisfaction relation can be verified by algorithms similar to ones for bisimilarity.

The following expected properties hold.

Proposition 3.1 *Let $\nu \subseteq Act_S$, $\mu \in Act_M$, $C \in CBh$, $s, s_i \in Sp$. Then,*

(1) $C \models_\nu \langle\mu\rangle s$
$\Leftrightarrow \exists(\mu', C').\ (C \xrightarrow{\mu'} C',\ \mu \doteq_\nu \mu',\ C' \models_\nu s)$

(2) $C \models_\nu [\mu]s$
$\Leftrightarrow \forall(\mu', C').\ ((C \xrightarrow{\mu'} C',\ \mu \doteq_\nu \mu') \Rightarrow C' \models_\nu s)$

(3) $C \models_\nu \bigwedge_{i\in I} s_i \ \Leftrightarrow\ \forall i \in I.\ C \models_\nu s_i$

(4) $C \models_\nu \bigvee_{i\in I} s_i \ \Leftrightarrow\ \exists i \in I.\ C \models_\nu s_i$

(5) $C \models_\nu K \ \Leftrightarrow\ \exists s.\ (C \models_\nu s,\ K \stackrel{\text{def}}{=} s)$ ∎

The following short notations are useful for expressing requirements for the passage of time:

$$s \vee_{\langle 0\rangle} t \equiv t,$$
$$s \vee_{\langle n+1\rangle} t \equiv s \vee \bigvee\{\langle\mu\rangle(s \vee_{\langle n\rangle} t) : \mu \in Act_M\},$$
$$s \vee_{[0]} t \equiv t,$$
$$s \vee_{[n+1]} t \equiv s \vee \bigwedge\{[\mu](s \vee_{[n]} t) : \mu \in Act_M\},$$
$$s \wedge_{\langle 0\rangle} t \equiv t,$$
$$s \wedge_{\langle n+1\rangle} t \equiv s \wedge \bigvee\{\langle\mu\rangle(s \wedge_{\langle n\rangle} t) : \mu \in Act_M\},$$
$$s \wedge_{[0]} t \equiv t,$$
$$s \wedge_{[n+1]} t \equiv s \wedge \bigwedge\{[\mu](s \wedge_{[n]} t) : \mu \in Act_M\}.$$

The specification $s \vee_{\langle n\rangle} t$ (resp. $s \vee_{[n]} t$) requires that s can be satisfied before n time-units pass or t can be satisfied after that, for some (resp. any) execution path. On the other hand, $s \wedge_{\langle n\rangle} t$ (resp. $s \wedge_{[n]} t$) requires that s is always satisfied until n time-units pass and t is satisfied after that, for some (resp. any) execution path.

The example of the researcher and the pizza-worker in Section 2 is used again. Since the pizza-worker can bake a pizza for 15 minutes and can drive and deliver for 10+1 minutes, the researcher can begin to eat in 26 minutes after an order as follows:

$$RP \models s_1 \equiv \langle order@bob, accept@john\rangle (\langle eat@bob\rangle\mathtt{tt} \vee_{\langle 27\rangle} \mathtt{ff}),$$

where $(\langle eat@bob\rangle\mathtt{tt} \vee_{\langle 27\rangle} \mathtt{ff})$ requires that the researcher '*bob*' can eat in 26 minutes because $\mathtt{ff}$ cannot be satisfied. However, since the pizza-worker can idle, the researcher can not always begin to eat it as follows:

$$RP \not\models s_2 \equiv \langle order@bob, accept@john\rangle (\langle eat@bob\rangle\mathtt{tt} \vee_{[27]} \mathtt{ff}).$$

where note the difference between $\vee_{\langle 27\rangle}$ in s_1 and $\vee_{[27]}$ in s_2. The specification s_1 implies that the researcher can eat before 26 minutes pass if the pizza-worker works hardly (i.e. for some execution path). On the other hand, s_2 implies that the researcher can eat before 26 minutes pass without respect to the pizza-worker (i.e. for any execution path).

Furthermore, the following satisfaction relation holds because the researcher never eats until 26 minutes pass.

$$RP \models \langle order@bob, accept@john\rangle ([eat@bob]\mathtt{ff} \wedge_{[26]} \mathtt{tt}).$$

Local specifications for the researcher *bob* can be easily verified by $\models_{\nu_{bob}}$, where $\nu_{bob} = \{a@bob : a \in \mathcal{N}_{ac}\}$. For example, it can be verified that the researcher can receive a pizza just after study for 30 minutes as follows: $RP \models_{\nu_{bob}} s_{bob}$, where

$$s_{bob} \equiv \langle order@bob\rangle\langle study@bob\rangle^{30}\langle receive@bob\rangle\mathtt{tt}$$

The satisfaction relation $\models_{\nu_{bob}}$ makes partial verification for *bob* possible by hiding actions of *john*. It is important to note that $RP \not\models s_{bob}$ because the pizza-worker *john* must bake the pizza and then drive while the researcher *bob* is studying if the behavior of *john* is considered.

4 Satisfiability

Before two agents combine, it is useful for avoiding unexpected behavior to verify consistency between their specifications. In *tLCA*, the consistency between s and t is replaced to *satisfiability* of $s \wedge t$ defined as follows.

Definition 4.1 *Let $s \in Sp$. s is* satisfiable *if and only if for some $C \in CBh$, $C \models s$.* ∎

In this section, we show that satisfiability in *tLCA* is undecidable by translating the membership problem in unrestricted grammars [9] into the satisfiability problem in *tLCA*. At first, action-sequence grammars G, a specification $GR_{(G)}$, and a concurrent behavior $\mathbf{CB}_{(G)}$ are defined for proving the undecidability.

Definition 4.2 *A tuple $G = \langle \mathcal{A}, a_0, \lambda\rangle$ is an* action-sequence grammar, *where $\mathcal{A} \subseteq \mathcal{N}_{ac}$ is a finite set of action-names, $a_0 \in \mathcal{N}_{ac}$ is the* initial action-name, *and λ is a finite set of* rewriting rules *such that: $\lambda \subseteq (\mathcal{A}^* - \{\varepsilon\}) \times \mathcal{A}^*$, where $\mathcal{A}^*$, ranged over by $\tilde{a}, \tilde{b}, \cdots$, is the set of finite action-sequences obtained by concatenating zero or more action-names in $\mathcal{A}$. The symbol ε is the empty action-sequence, thus $\tilde{a}\varepsilon = \varepsilon\tilde{a} = \tilde{a}$.* ∎

Definition 4.3 *Let $G = \langle \mathcal{A}, a_0, \lambda\rangle$ be an action-sequence grammar. The* G-language *$\mathcal{L}(G)$ is defined as follows: $\mathcal{L}(G) = \bigcup_{n\geq 0} \mathcal{L}^{(n)}(G)$, $\mathcal{L}^{(0)}(G) = \{a_0\}$, and $\mathcal{L}^{(n+1)}(G) = \{\tilde{a}_1\tilde{b}\tilde{a}_2 : \exists\tilde{a}.\ \tilde{a}_1\tilde{a}\tilde{a}_2 \in \mathcal{L}^{(n)}(G), (\tilde{a}, \tilde{b}) \in \lambda\}$.* ∎

By Definition 4.2, action-sequence grammars are unrestricted grammars because there is no restriction on rewriting rules. It is known that the membership problem ($\tilde{a} \in \mathcal{L}(G)$ or $\tilde{a} \notin \mathcal{L}(G)$?) in such grammars is undecidable.

Next, we define a finite-state specification $GR_{(G)}$ for each G.

Definition 4.4 *Let $\mathtt{p}, \mathtt{q} \in \mathcal{N}_{ag}$ and $G = \langle \mathcal{A}, a_0, \lambda\rangle$ be an action-sequence grammar (note: $\mathtt{p}, \mathtt{q}$ are not variables but instances). The finite-state specification $GR_{(G)}$ is defined from G as follows:*

$$GR_{(G)} \stackrel{\text{def}}{=} RW_{(G)} \wedge EQ^{(\mathtt{q},\mathtt{p})}_{(G)} \wedge \partial\langle a_0@\mathtt{p}\rangle\partial\langle \mathtt{end}@\mathtt{p}\rangle\mathtt{tt}$$

$$RW_{(G)} \stackrel{\text{def}}{=} \bigwedge\{[\tilde{a}@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)} : (\tilde{a}, \tilde{b}) \in \lambda\} \cup \{\partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle RW_{(G)} : a \in \mathcal{A}_{\mathtt{end}}\}$$

$$EQ^{(\psi,\varphi)}_{(G)} \stackrel{\text{def}}{=} \bigwedge\{\partial[a@\psi]\partial\langle a@\varphi\rangle EQ^{(\psi,\varphi)}_{(G)} : a \in \mathcal{A}_{\mathtt{end}}\}$$

where $\mathcal{A}_{\mathtt{end}} = \mathcal{A} \cup \{\mathtt{end}\}$, $\mathtt{end} \in \mathcal{N}_{ac}$, $\mathtt{end} \notin \mathcal{A}$, and specifications $\langle\tilde{a}@\psi\rangle^* s$ and $[\tilde{a}@\psi]^* s$ are short notations defined as follows:

$$\langle\varepsilon@\psi\rangle^* s \equiv s, \quad \langle a\tilde{a}'@\psi\rangle^* s \equiv \partial\langle a@\psi\rangle\langle\tilde{a}'@\psi\rangle^* s,$$

$$[\varepsilon@\psi]^* s \equiv s, \quad [a\tilde{a}'@\psi]^* s \equiv \partial[a@\psi][\tilde{a}'@\psi]^* s$$

Furthermore, for any specification s, a Constant ∂s is defined as follows: $\partial s \stackrel{\text{def}}{=} s \wedge [\emptyset]\partial s$. ∎

The part $EQ^{(\mathtt{q},\mathtt{p})}_{(G)}$ copies execution-sequences of agent-$\mathtt{q}$ into agent-$\mathtt{p}$. And $RW_{(G)}$ copies execution-sequences of agent-$\mathtt{p}$ into agent-$\mathtt{q}$, rewriting according to rules $(\tilde{a}, \tilde{b}) \in \lambda$. The Constant ∂s is necessary

for waiting while another agent performs actions. Consequently, $GR_{(G)}$ requires agents to perform all the action-sequences contained in $\mathcal{L}(G)$ as shown in the following lemma.

Lemma 4.1 *If $C \models GR_{(G)}$ and $\tilde{a} \in \mathcal{L}(G)$, then for some C', $C \stackrel{\{\tilde{a}\,\mathtt{end@p}\}}{\Longrightarrow} C'$, where $C \stackrel{\{a_1 \cdots a_n @\psi\}}{\Longrightarrow} C'$ if and only if $C\ (\stackrel{\emptyset}{\longrightarrow})^*\ \stackrel{\{a_1@\psi\}}{\longrightarrow}\ (\stackrel{\emptyset}{\longrightarrow})^* \cdots (\stackrel{\emptyset}{\longrightarrow})^*\ \stackrel{\{a_n@\psi\}}{\longrightarrow}\ (\stackrel{\emptyset}{\longrightarrow})^*\ C'$, where $(\stackrel{\emptyset}{\longrightarrow})^*$ is the passage of zero or more time-units.* ∎

Then, a concurrent behavior $\mathbf{CB}_{(G)}$ is defined for each G.

Definition 4.5 *Let $\mathtt{p}, \mathtt{q} \in \mathcal{N}_{ag}$ and $G = \langle \mathcal{A}, a_0, \lambda \rangle$ be an action-sequence grammar. Then, the concurrent behavior $\mathbf{CB}_{(G)}$ is defined as follows*:

$$\mathbf{CB}_{(G)} \equiv \mathbf{Bh}_{(G)}@\mathtt{p} \mid \mathbf{Bh}_{(G)}@\mathtt{q},$$
$$\mathbf{Bh}_{(G)} \equiv \textstyle\sum_{n\geq 0} \mathbf{Bh}_{(G,n)},$$
$$\mathbf{Bh}_{(G,0)} \equiv a_0.\mathtt{end}.\mathbf{I},$$
$$\mathbf{Bh}_{(G,n+1)} \equiv \mathbf{Rw}_{(G)}(\mathbf{Bh}_{(G,n)}),$$

where $\mathbf{Rw}_{(G)}(P)$ is defined as follows:

$$\mathbf{Rw}_{(G)}(P) \equiv \textstyle\sum\{\tilde{b}.P' : P \stackrel{\tilde{a}}{\longrightarrow} P', (\tilde{a}, \tilde{b}) \in \lambda\} \cup \{a.\mathbf{Rw}_{(G)}(P') : P \stackrel{a}{\longrightarrow} P'\}$$

where $\tilde{a}.P$ is defined as: $\varepsilon.P \equiv P$, $a\tilde{b}.P \equiv a.\tilde{b}.P$. And $P \stackrel{\varepsilon}{\longrightarrow} P'$ represents $P \equiv P'$, and $P \stackrel{a\tilde{a}'}{\longrightarrow} P'$ represents $P \stackrel{a}{\longrightarrow} P'' \stackrel{\tilde{a}'}{\longrightarrow} P'$ for some P''. ∎

Lemma 4.2 shows that any action-sequence executed by $\mathbf{CB}_{(G)}$ is contained in $\mathcal{L}(G)$. And Lemma 4.3 shows that $\mathbf{CB}_{(G)}$ satisfies $GR_{(G)}$.

Lemma 4.2 *If $\mathbf{CB}_{(G)} \stackrel{\{\tilde{a}\,\mathtt{end@p}\}}{\Longrightarrow} C'$, then $\tilde{a} \in \mathcal{L}(G)$.* ∎

Lemma 4.3 $\mathbf{CB}_{(G)} \models GR_{(G)}$ ∎

Finally, the following result is obtained from Lemmas 4.1, 4.2, and 4.3.

Theorem 4.4 *Satisfiability in the process logic tLCA is undecidable.*

Proof Let $G = \langle \mathcal{A}, a_0, \lambda \rangle$ be an action-sequence grammar. Then, we show that the following relation holds because $\tilde{a} \notin \mathcal{L}(G)$ is undecidable.

$(GR_{(G)} \wedge [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff})$ is satisfiable $\Leftrightarrow \tilde{a} \notin \mathcal{L}(G)$

If part ($\Rightarrow$): By the assumption, for some C, $C \models (GR_{(G)} \wedge [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff})$. Now assume that $\tilde{a} \in \mathcal{L}(G)$. Thus, by Lemma 4.1, $C \stackrel{\{\tilde{a}\,\mathtt{end@p}\}}{\Longrightarrow} C'$ because $C \models GR_{(G)}$. However, this contradicts the assumption $C \models [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff}$. Hence, $\tilde{a} \notin \mathcal{L}(G)$.

If part ($\Leftarrow$): We show the contraposition. Assume that $(GR_{(G)} \wedge [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff})$ is not satisfiable. By Lemma 4.3, $\mathbf{CB}_{(G)} \models GR_{(G)}$. Therefore, $\mathbf{CB}_{(G)} \not\models [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff}$ because if $\mathbf{CB}_{(G)} \models [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff}$ then $\mathbf{CB}_{(G)} \models (GR_{(G)} \wedge [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff})$ and this contradicts the assumption. This unsatisfaction ($\mathbf{CB}_{(G)} \not\models [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff}$) implies that for some C', $\mathbf{CB}_{(G)} \stackrel{\{\tilde{a}\,\mathtt{end@p}\}}{\Longrightarrow} C'$. Hence, by Lemma 4.2, $\tilde{a} \in \mathcal{L}(G)$. ∎

5 A Process Algebra *tICCA*

In this paper, our purpose is to show how to avoid unexpected behavior caused by composition of agents. For example, suppose that agent-ψ and agent-φ behave like P and Q (described in *tCCA*), and must satisfy s and t (described in *tLCA*), respectively. Then, it is a useful method for avoiding unexpected behavior to verify the consistency between s and t before agent-ψ and agent-φ combine. However, there is no algorithm for verifying the consistency as shown in Theorem 4.4. To solve this problem, although it is expected to use a decidable and useful subclass of *tLCA*, we have not been able to define such subclass yet.

The other solution is to directly verify whether $P@\psi|Q@\varphi$ satisfies $s \wedge t$ (i.e. $P@\psi|Q@\varphi \models s \wedge t$), or not. There are algorithms for verifying the satisfaction relation $\models$. However, the number of reachable states of concurrent behavior explosively increases with concurrency level. To decrease the number of states, it is useful to hide useless actions for verification because verification often considers only some actions (not all actions). Therefore we propose to use a process algebra *tICCA* for expressing *intermediate specifications* (also called *im-specifications*) between concurrent behaviors (*tCCA*) and specifications (*tLCA*). In *tICCA*, the number of reach-

able states can decreases by hiding uninteresting actions. In such approach, it is important to know which actions can be hidden. In this section, we show how to verify composition of agents in *tICCA* instead of in *tCCA*.

First, it is assumed that a set $\mathcal{K}_{im}$, ranged over by $K, \cdots$, of *im-Constants* is given. Then, the set *ISp*, ranged over by $S, T, \cdots$, of *im-specifications* is the smallest set which contains the following expressions:

K : Recursion ($K \in \mathcal{K}_{im}$),
$\mathbf{0}$: Stop,
$\mu;S$: (Insistent) Prefix ($\mu \in Act_M$),
$\bigoplus_{i\in I} S_i$: (Weak) Summation

where S, S_i, T are already in *ISp*. These operators have the binding power such that: Prefix > Summation.

It is important that the Prefix is not $a.S$ but $\mu;S$, comparing with *tCCA*. The multi-action μ is directly prefixed to S, and the execution of μ is not delayed (i.e. $\mu;S \stackrel{\emptyset}{\not\longmapsto} \mu;S$). For Summation $\bigoplus_{i\in I} S_i$, the choice is made even by a time passage $\stackrel{\emptyset}{\longmapsto}$. This operator is similar to 'weak choice' in TCCS [15]. We use a short notation for binary selection as follows: $S_1 \oplus S_2 \equiv \bigoplus_{i\in\{1,2\}} S_i$. Furthermore, it is assumed that for every im-Constant $K \in \mathcal{K}_{im}$, there is a defining equation $K \stackrel{\text{def}}{=} S$ for some $S \in ISp$, and also that an idling im-Constant for each $S \in ISp$ is defined as follows: $\partial S \stackrel{\text{def}}{=} S \oplus \emptyset;\partial S$. Although the symbol ∂ has already been used for $s \in Sp$ in Definition 4.4, we use the same symbol ∂ for s and S because their function is the same.

Semantics of im-specifications is given by the labelled transition system $\langle ISp, Act_M, \{\stackrel{\mu}{\longmapsto}: \mu \in Act_M\}\rangle$, where the set $\stackrel{\mu}{\longmapsto}$ of transitions is the smallest relation satisfying the rules in Fig.5.

Then, we define im-Constants $S|T$, $S\backslash M$, and S/ν, for each $S, T \in ISp$, $M \subseteq Act_M$, and $\nu \subseteq Act_S$ as follows:

Name	Hypothesis	⇒	Conclusion
I.Act		⇒	$\mu;S \stackrel{\mu}{\longmapsto} S$
W.Sum	$\exists i \in I.\ S_i \stackrel{\mu}{\longmapsto} S'$	⇒	$\bigoplus_{i\in I} S_i \stackrel{\mu}{\longmapsto} S'$
Rec	$K \stackrel{\text{def}}{=} S,\ S \stackrel{\mu}{\longmapsto} S'$	⇒	$K \stackrel{\mu}{\longmapsto} S'$

Fig.5 Inference rules for $\stackrel{\mu}{\longmapsto} \subseteq ISp \times ISp$

$$S|T \stackrel{\text{def}}{=} \bigoplus\{(\mu\cup\nu);(S'|T') : S \stackrel{\mu}{\longmapsto} S', T \stackrel{\nu}{\longmapsto} T'\}$$
$$S\backslash M \stackrel{\text{def}}{=} \bigoplus\{\mu;(S'\backslash M) :\ S \stackrel{\mu}{\longmapsto} S',\ \mu \in M \cup \{\emptyset\}\}$$
$$S/\nu \stackrel{\text{def}}{=} \bigoplus\{(\mu\cap\nu);(S'/\nu) :\ S \stackrel{\mu}{\longmapsto} S'\}$$

Next, the satisfaction relation between a concurrent behavior and an im-specification is defined.

Definition 5.1 *Let $\nu \subseteq Act_S$. A set $\mathcal{R} \subseteq CBh \times ISp$ is a* (ν)-im-satisfaction-subset, *if $(C, S) \in \mathcal{R}$ implies that the following conditions (i) and (ii) hold for every $\mu \in Act_M$,*

(i) if $C \stackrel{\mu}{\longrightarrow} C'$, then for some S',
$S \stackrel{\mu\cap\nu}{\longmapsto} S'$ and $(C', S') \in \mathcal{R}$,

(ii) if $S \stackrel{\mu}{\longmapsto} S'$, then for some μ' and S',
$\mu = \mu' \cap \nu$, $C \stackrel{\mu'}{\longrightarrow} C'$, and $(C', S') \in \mathcal{R}$.

Then, if $(C, S) \in \mathcal{R}$ for some (ν)-im-satisfaction-subset $\mathcal{R}$, then C satisfies *S with respect to ν, written $C \vdash_\nu S$. For a special case, $C \vdash_{Act_S} S$ is written $C \vdash S$. The bisimilarity $\sim$ for $S \sim T$ is defined in the same way as $C \vdash S$.* ∎

By hiding uninteresting single-actions $a@\psi \notin \nu$, the number of states of im-specifications can decrease. The set ν of single-actions in $\vdash_\nu$ is also called an *available set*, similarly to the case of $\models_\nu$. For example, $C_{abc} \equiv a.b.\mathbf{I}@\psi|c.\mathbf{I}@\varphi$ satisfies the following S_{abc} and S_b with respect to the available sets $\nu_1 = \{a@\psi, b@\psi, c@\varphi\}$ and $\nu_2 = \{b@\psi\}$, respectively (i.e. $C_{abc} \vdash_{\nu_1} S_{abc}$ and $C_{abc} \vdash_{\nu_2} S_b$).

$$\begin{aligned} S_{abc} \equiv \partial(&\{a@\psi\};\partial(\{b@\psi\};\partial\{c@\varphi\};\partial\mathbf{0} \\ &\oplus\{c@\varphi\};\partial\{b@\psi\};\partial\mathbf{0} \\ &\oplus\{b@\psi, c@\varphi\};\partial\mathbf{0}) \\ \oplus&\{c@\varphi\};\partial\{a@\psi\};\partial\{b@\psi\};\partial\mathbf{0} \\ \oplus&\{a@\psi, c@\varphi\};\partial\{b@\psi\};\partial\mathbf{0}) \end{aligned}$$

$$S_b \equiv \partial\emptyset;\partial\{b@\psi\};\partial\mathbf{0}$$

The transition graphs of S_{abc} and S_b are shown in

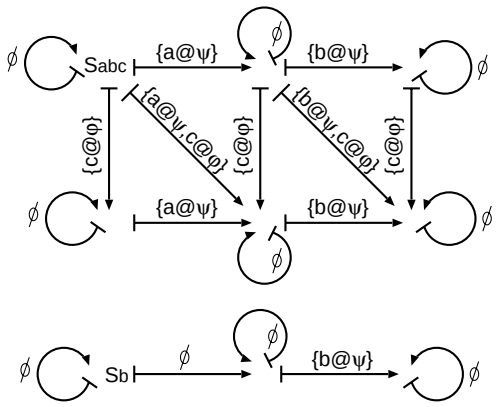

Fig.6 The transition graphs of S_{abc} and S_b

Fig.6. The im-specification S_b concisely expresses behavior with respect to the single-action $b@\psi$.

The following Propositions 5.1 and 5.2 show how the (ν)-satisfaction relation $\vdash_\nu$ is preserved by the operators $|$ and $\setminus$ of concurrent behaviors. The condition ($\mathtt{res}(act(C), M, \nu) \subseteq M$) in Proposition 5.2 is a sufficient condition for preserving the (ν)-satisfaction relation $\vdash_\nu$ when the concurrent behavior C is restricted within M. In order to gain the sufficient and necessary condition, since transitions in C must be considered, we present the simple (but useful) sufficient condition which considers only actions in C. Intuitively, the condition ($\mathtt{res}$) requires that restricted single-actions must be included in the available set ν.

Proposition 5.1 *If $Agn(C_1) \cap Agn(C_2) = \emptyset$ and $C_1 \vdash_{\nu_1} S_1$ and $C_2 \vdash_{\nu_2} S_2$, then $C_1|C_2 \vdash_\nu S_1|S_2$, where $\nu = (\nu_1 \downarrow Agn(C_1)) \cup (\nu_2 \downarrow Agn(C_2))$, where $\nu\downarrow\Psi$ is a subset of ν defined as follows: $\nu\downarrow\Psi = \{a@\psi \in \nu : \psi \in \Psi\}$.* ∎

Proposition 5.2 *If $\mathtt{res}(act(C), M, \nu) \subseteq M$ and $C \vdash_\nu S$, then $C \setminus M \vdash_\nu S \setminus M'$, where $M' = \{\mu \cap \nu : \mu \in M\}$, the function $\mathtt{res} : 2^{Act_M} \times 2^{Act_M} \times 2^{Act_S} \to 2^{Act_M}$ is defined as follows:*

$$\mathtt{res}(M_0, M, \nu) = \{\mu_0 \in M_0 : \exists \mu \in M \cup \{\emptyset\}. \ \mu_0 \doteq_\nu \mu, \mu_0 \neq \emptyset\},$$

and the function $act : CBh \to 2^{Act_M}$ is defined as follows: $act(P@\psi) = \{\{a@\psi\} : a \in acn(P)\} \cup \{\emptyset\}$, $act(C_1|C_2) = \{\mu_1 \cup \mu_2 : \mu_1 \in act(C_1), \mu_2 \in act(C_2)\}$, $act(C \setminus M) = (act(C) \cap M) \cup \{\emptyset\}$, where $acn(P)$ is a set of action-names occurring in P. Thus, $act(C)$ is the set of feasible multi-actions in C at most. ∎

The following Proposition 5.3 is used for hiding uninteresting actions, in other words, for obtaining a more abstract im-specification S/μ than S. Furthermore, Proposition 5.4 shows that $C \models_\nu s$ can be verified by $S \models_\nu s$ if $C \vdash_\nu S$, where $S \models_\nu s$ is defined by replacing $\xrightarrow{\mu}$ in Definition 3.1 with $\overset{\mu}{\longmapsto}$. Thus these propositions mean that verification-cost of $C \models_\nu s$ can decrease.

Proposition 5.3 *If $C \vdash_\nu S$, then for any $\mu \subseteq Act_S$, $C \vdash_{\nu\cap\mu} S/\mu$.* ∎

Proposition 5.4 *Assume that $C \vdash_\nu S$. Then, $C \models_\nu s$ if and only if $S \models_\nu s$.* ∎

The example of the researcher and the pizza-worker in Section 2 is used again. Now, assume that the available sets are given as follows:
ν_r=$\{order@bob, receive@bob, cancel@bob\}$,
ν_p=$\{accept@john, deliver@john, canceled@john\}$
Then, an im-specification S_r of the researcher $RES@bob$ with respect to ν_r and an im-specification S_p of the pizza-worker $PIZ@john$ with respect to ν_p are given as follows (i.e. $RES@bob \vdash_{\nu_r} S_r$ and $PIZ@john \vdash_{\nu_p} S_p$):

$$\begin{aligned} S_r &\equiv \partial\{order@bob\};S^{(60)}_{wait}, \\ S^{(n+1)}_{wait} &\equiv \emptyset;S^{(n)}_{wait} \oplus \{receive@bob\};\partial\mathbf{0}, \\ S^{(0)}_{wait} &\equiv \partial\{cancel@bob\};\partial\mathbf{0}, \end{aligned}$$

$$\begin{aligned} S_p &\equiv \partial\{accept@john\};S^{(25)}_{work}, \\ S^{(n+1)}_{work} &\equiv \partial(\emptyset;S^{(n)}_{work} \oplus S_c), \\ S^{(0)}_{work} &\equiv \partial(\{deliver@john\};\partial S_c \oplus S_c), \\ S_c &\equiv \{canceled@john\};\partial\mathbf{0}. \end{aligned}$$

Then, Propositions 5.1 and 5.2 show that $RP \equiv (RES@bob|PIZ@john) \setminus M_{rp} \vdash_{\nu_r\cup\nu_p} (S_r|S_p) \setminus M'_{rp}$ because $\mathtt{res}(act(RES@bob|PIZ@john), M_{rp}, \nu_r \cup \nu_p) \subseteq M_{rp}$, where $M'_{rp} = M_{sync} \cup \{\emptyset\}$. Furthermore, by Proposition 5.3, $RP \vdash_{\nu_r} (S_r|S_p)\setminus M'_{rp}/\nu_r$.

Here, if S'_r is given as follows, then we can show that $S'_r \sim (S_r|S_p)\backslash M'_{rp}/\nu_r$.

$$S'_r \equiv \partial\{order@bob\};S^{(60,25)}_{wait'},$$
$$S^{(n+1,m+1)}_{wait'} \equiv \emptyset;S^{(n,m)}_{wait'} \oplus \emptyset;S^{(n,m+1)}_{wait'},$$
$$S^{(n+1,0)}_{wait'} \equiv \emptyset;S^{(n,0)}_{wait'} \oplus \{receive@bob\};\partial\mathbf{0},$$
$$S^{(0,m)}_{wait'} \equiv \partial\{cancel@bob\};\partial\mathbf{0}.$$

This implies that $RP \vdash_{\nu_r} S'_r$.

Although both of S_r and S'_r are im-specifications with respect to the researcher *bob* (i.e. ν_r), S_r shows an abstract behavior of the independent researcher, and S'_r shows an abstract behavior of the researcher dependent with the pizza-worker. For example, S_r shows that the researcher can immediately receive a pizza after the order because S_r does not consider to make pizza. On the other hand, S'_r shows that the researcher cannot receive a pizza until 25 minutes pass because the pizza-worker must bake it for 15 minutes and drive for 10 minutes before the delivery.

Now, we consider how to verify $RP \models_{\nu_r} s_r$, where $s_r \equiv \langle order@bob\rangle(\langle receive@bob\rangle\mathtt{tt} \vee_{\langle 26\rangle} \mathtt{ff})$. At first, we can prove that $S'_r \models_{\nu_r} s_r$ according to Definition 3.1. Hence, by Proposition 5.4, we have that $RP \models_{\nu_r} s_r$ because $RP \vdash_{\nu_r} S'_r$. The verification-cost of $S'_r \models_{\nu_r} s_r$ is lower than one of $RP \models_{\nu_r} s_r$ because the numbers of states of S'_r and RP are 390 and 34,991, respectively.

6 Related Work

Many process algebras based on non-interleaving semantics have been proposed (e.g. in [1][4][8][14]) by considering locality or causality between actions, and process logics with locality have also been given (e.g. in [1]). Also, many timed process algebras such as TCCS[15] or TPL[5] have been proposed, and a uniform framework has also been proposed for extending arbitrary process languages with discrete time[20]. Furthermore, a timed process algebra which has durational actions has been proposed and a process logic for the process algebra has been presented[3]. However, satisfiability in such process logics has not been discussed.

Many temporal logics for non-interleaving traces (also called Mazurkiewicz's traces[11]) have been proposed (e.g. in [6][10][16][19]), and satisfiability has been studied. However they have not considered the passage of time. In these non-interleaving non-timed temporal logics, a kind of commutativity of concurrent requirements holds. For example, if time is not considered, two specifications $\langle a@\psi\rangle\langle b@\varphi\rangle\mathtt{tt}$ and $\langle b@\varphi\rangle\langle a@\psi\rangle\mathtt{tt}$ are equal because agent-φ does not alter its own state while agent-ψ performs action-a. In [10][17], satisfiability problems in several non-interleaving temporal logics can be translated into a *recurring tiling problem* which has been shown to be undecidable. The essential property for the translation is the commutativity of concurrent requirements as mentioned above.

On the other hand, the commutativity of concurrent requirements does not hold in *tLCA*. For example, $\langle a@\psi\rangle\langle b@\varphi\rangle\mathtt{tt}$ and $\langle b@\varphi\rangle\langle a@\psi\rangle\mathtt{tt}$ are not equal in *tLCA* because agent-ψ consumes one time-unit for performing the action-a and agent-φ can alter its own state in the passage of time. Hence, undecidability in *tLCA* cannot be directly proven from the results in [10][17]. Furthermore, it is important to note that we have proven undecidability in *tLCA without Disjunction and Eventuality*, as shown in the specification $(GR_{\langle G\rangle} \wedge [\tilde{a}\,\mathtt{end@p}]^*\mathtt{ff})$ in the proof of Theorem 4.4. Disjunction and Eventuality were necessary for proving undecidability in [10][17].

7 Discussion

In this section, we discuss our approach.

Automatic generation of im-specifications: Designers of agents need not give im-specifications because im-specifications (in *tICCA*) can be automatically generated from concurrent behaviors (in

tCCA). For each concurrent behavior $C \in CBh$, a *full* im-specification $\mathtt{imsp}(C)$ of C with respect to Act_S (i.e. $C \vdash \mathtt{imsp}(C)$) is easily given as follows:

$$\mathtt{imsp}(C) \stackrel{\text{def}}{=} \bigoplus\{\mu;\mathtt{imsp}(C') : \ C \stackrel{\mu}{\longrightarrow} C'\}.$$

Then, for each available set $\nu \subseteq Act_S$, $\mathtt{imsp}(C)/\nu$ is an im-specification of C with respect to ν by Proposition 5.3. Furthermore, the *reduced normal form* $\mathtt{imsp}_\nu(C)$ of $\mathtt{imsp}(C)/\nu$ can be obtained by the algorithms in [7], where the reduced normal form is defined as follows: an im-specification S_{min} is the reduced normal form of S if the following condition is satisfied: for every S' such that $S' \sim S$, $\|S_{min}\| \leq \|S'\|$, where $\|S\|$ is the number of reachable states from S.

For example, when a behavior P of agent-ag is given, im-specifications of agent-ag with respect to several available sets ν_i $(i \in I)$ are generated from $P@ag$ (i.e. $\mathtt{imsp}_{\nu_i}(P@ag)$) and are saved in agent-ag. If agent-ag combines with the other agent under some restrictions, then an im-specification which satisfies the condition `res` in Proposition 5.2 is selected and used for verification.

Non-interleaving: We have introduced non-interleaving for expressing concurrency because there is not always a *concurrent* system to satisfy a given specification even if there is a *sequential* system to satisfy it. For example, if non-interleaving is not considered, then a specification $s \equiv \langle a\rangle\langle b\rangle\mathtt{tt} \wedge [\emptyset][b]\mathtt{ff}$ is satisfiable (e.g. $a.b.\mathbf{I} \models s$). However, if it is required that a and b are performed by different agents like $s' \equiv \langle a@\psi\rangle\langle b@\varphi\rangle\mathtt{tt} \wedge [\emptyset][b@\varphi]\mathtt{ff}$, then s' is not satisfiable. Non-interleaving is necessary for correctly checking whether there is a *concurrent* system to satisfy a given specification, or not.

Durational actions: We have assumed that the passage of one time-unit is always needed for executing any action. If this assumption is removed, notions of non-interleaving and time can be *separately* discussed, and undecidability is proven more easily than one in *tLCA*. However, there are two reasons why we assume it. One is to remove unnatural situations such that actions can infinitely be performed in a finite time. Another is to clarify decidability under the assumption because we prospected that *tLCA* might be decidable by the assumption (i.e. the restriction) because the commutativity of concurrent requirements as mentioned in Section 6 does not hold. It is a future work to define a decidable and useful subclass of *tLCA*.

Expressive power of *tLCA* and *tICCA*: As shown in the following proposition, specifications in *Sp* can not distinguish between two concurrent behaviors in *CBh* if and only if im-specifications in *ISp* can not distinguish between them.

Proposition 7.1 *Let $\nu \subseteq Act_S$, $C, D \in CBh$. Then*

$$Sp_\nu(C) = Sp_\nu(D) \iff ISp_\nu(C) = ISp_\nu(D)$$

where $Sp_\nu(C) = \{s \in Sp : C \models_\nu s\}$ and $ISp_\nu(C) = \{S \in ISp : C \vdash_\nu S\}$. ∎

Errors of clocks: Although *tCCA* assumes that each agent has an exact local clock, errors of local clocks can be simulated by nondeterminism as follows: $a(n\pm1).P \equiv a(n-1).P + a(n).P + a(n+1).P$. By verifying agents in such situation with errors, more robust agent systems can be constructed.

A tool: We are now implementing a prototype of verification tool for *tCCA*, *tLCA*, and *tICCA* in Java. We have already implemented parts of analyzing syntax and inferring transitions. Fig.7 shows a screen shot of the prototype, where the number of reachable states and the transitions of S'_τ (in Section 5) and *RP* (in Section 2) are computed, although symbols in the tool is slightly different from ones of *tCCA* and *tICCA*. We plan to implement functions for verifying satisfaction relations $C \models_\nu s$ and $C \vdash_\nu S$, and for generating im-specifications from concurrent behaviors.

8 Conclusion

We have defined *tCCA* for describing concurrent behaviors of composite agents. *tLCA* has also been

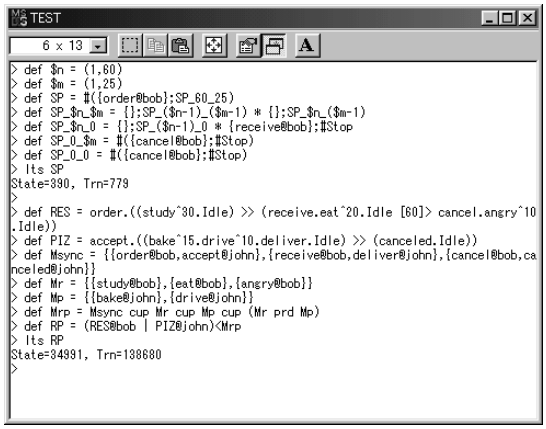

Fig.7 A screen shot of the prototype

defined for describing specifications and it has been proven that satisfiability in *tLCA* is undecidable. Then we have proposed to use intermediate specifications described in *tICCA* instead of directly checking consistency between specifications, and then have shown useful propositions for verifying composition of agents.

tCCA cannot express *mobility* such as π-calculus [13] or mobile ambient [2]. An extension of *tCCA* with such mobility is a future work.

Acknowledgments

The authors wish to express our gratitude to Professor Shinichi Honiden of National Institute of Informatics, and to express our gratitude to Instructor Tadashi Iijima of Keio University. They are also members of ESP (Evolutionary System Project). The project ESP has been done with the support of JSPS(Japan Society for the Promotion of Science) Grants-in-Aid for Scientific Research.

References

[1] Boudol, G., Castellani, I., Hennessy, M., and Kiehn, A.: Observing localities, *Theoretical Computer Science*, Vol.114, pp.31-61, 1993.

[2] Cardelli, L. and Gordon, A.D.: Mobile Ambient, FoSSaCS'98, Lecture Notes in Computer Science 1378, Springer-Verlag, pp.140-155, 1998.

[3] Chen, X.J. and Corradini, F.: On the Specification and Verification of Performance Properties for a Timed Process Algebra, AMAST'97, Lecture Notes in Computer Science 1349, pp.123-137, 1997.

[4] Degano, P. and Priami, C.: Non-interleaving semantics for mobile processes, *Theoretical Computer Science*, Vol.216, pp.237-270, 1999.

[5] Hennessy, M. and Regan, T.: A process algebra for timed systems, *Information and computation*, Vol.117, pp.221-239, 1995.

[6] Huhn, M., Niebert, P., and Wallner, F.: Verification Based on Local States, TACAS'98, Lecture Notes in Computer Science 1384, pp.36-51,1998.

[7] Kanellakis, P.C. and Smolka, S.A.: CCS Expressions, Finite State Processes, and Three Problems of Equivalence, *Information and computation*, Vol.86, No.1, pp.43-68, 1990.

[8] Krishnan, P.: Distributed CCS, CONCUR'91, Lecture Notes in Computer Science 527, pp.393-407, 1991.

[9] Linz, P.: *An Introduction to Formal Languages and Automata*, Jones and Bartlett Publishers, 1990.

[10] Lodaya, K., Parikh, R., Ramanujam, R., and Thiagarahan, P.S.: A Logical Study of Distributed Transition Systems, *Information and computation*, Vol.119, pp.91-118, 1995.

[11] Mazurkiewicz, A.: Basic Notions of Trace Theory, Lecture Notes in Computer Science 354, pp.285-363, 1988.

[12] Milner, R.: *Communication and Concurrency*, Prentice-Hall, 1989.

[13] Milner, R., Parrow, J., and Walker, D.: A Calculus of Mobile Processes, parts I and II, *Information and Computation*, Vol.100, No.1, pp.1-40 and 41-77, 1992.

[14] Montanari, U. and Yankelevich, D.: A Parametric Approach to Localities, ICALP'92, Lecture Notes in Computer Science 623, Springer-Verlag, pp.617-628, 1992.

[15] Moller, F. and Tofts, C.: A Temporal Calculus of Communicating Systems, CONCUR'90, Lecture Notes in Computer Science 458, Springer-Verlag, pp.401-415, 1990.

[16] Niebert, P.: A ν-Calculus with Local Views for Systems of Sequential Agents, MFCS'95, Lecture Notes in Computer Science 969, pp.563-573, 1995.

[17] Penczek, W.: Undecidability of Propositional Temporal Logics on Trace Systems, Information Processing Letters 43, pp.147-153, 1992.

[18] Stirling, C.: An Introduction to Modal and Temporal Logics for CCS, Concurrency: Theory, Language, and Architecture, Lecture Notes in Computer Science 491, Springer-Verlag, pp.2-20, 1989.

[19] Thiagarahan, P.S.: A Trace Based Extension of Linear Time Temporal Logic, Proc. of IEEE LICS, pp.438-447, 1994.

[20] Ulidowski, I. and Yuen, S.: Extending Process Languages with Time, AMAST'97, Lecture Notes in Computer Science 1349, pp.525-538, 1997.

A Appendix

Omitted proofs of lemmas and propositions in this paper are given.

Proof of Proposition 3.1

(1) Since $DC(\langle\mu\rangle s) = \{\{\langle\mu\rangle s\}\}$, $C \models_\nu \langle\mu\rangle s$ if and only if for some C' and μ', $C \xrightarrow{\mu'} C'$, $\mu \doteq_\nu \mu'$, and $C' \models_\nu s$ by (i) in Definition 3.1.

(2) Since $DC([\mu]s) = \{\{[\mu]s\}\}$, $C \models_\nu [\mu]s$ if and only if for any C' and μ' such that $C \xrightarrow{\mu'} C'$ and $\mu \doteq_\nu \mu'$, $C' \models_\nu s$ by (ii) in Definition 3.1.

(3) $(\Rightarrow)$ Assume that $C \models_\nu \bigwedge_{i\in I} s_i$. Thus, for some $\sigma \in DC(\bigwedge_{i\in I} s_i)$, the pair (C, σ) satisfies the conditions (i) and (ii) in Definition 3.1. Here, for each $i \in I$, for some $\sigma_i \in DC(s_i)$, $\sigma = \bigcup_{i\in I} \sigma_i$ by the definition of $DC(\bigwedge_{i\in I} s_i)$. For every $i \in I$, since the pair (C, σ_i) satisfies (i) and (ii), we have $C \models_\nu s_i$.

$(\Leftarrow)$ We can reverse the proof of $(\Rightarrow)$.

(4) $(\Rightarrow)$ Assume that $C \models_\nu \bigvee_{i\in I} s_i$. Thus, for some $\sigma \in DC(\bigvee_{i\in I} s_i)$, the pair (C, σ) satisfies the conditions (i) and (ii) in Definition 3.1. Here, for some $i \in I$, $\sigma \in DC(s_i)$ because $DC(\bigvee_{i\in I} s_i) = \bigcup_{i\in I} DC(s_i)$. Hence, for some $i \in I$, $C \models_\nu s_i$.

$(\Leftarrow)$ We can reverse the proof of $(\Rightarrow)$.

(5) This is trivial because $DC(K) = DC(s)$ if $K \stackrel{\text{def}}{=} s$. ∎

Proof of Lemma 4.1

Let $G = \langle \mathcal{A}, a_0, \lambda\rangle$ be an action-sequence grammar. If $C \models GR_{(G)}$, then it can be assumed that C has the form $P@\mathtt{p}|Q@\mathtt{q}$ without losing generality because $GR_{(G)}$ contains only requirements of the form $\langle a@\mathtt{p}\rangle s$, $[a@\mathtt{p}]s$, or $[\emptyset]s$, for two agents-$\mathtt{p}$ and $\mathtt{q}$. For proving Lemma 4.1, we prove that if $P@\mathtt{p}|Q@\mathtt{q} \models GR_{(G)}$, then the following sub-lemmas hold.

(4.1.1) $P@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle a_0\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$,

(4.1.2) If $P@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}_1\tilde{a}\tilde{a}_2\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ and $(\tilde{a}, \tilde{b}) \in \lambda$, then $P@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}_1\tilde{b}\tilde{a}_2\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$,

(4.1.3) If $P@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, then $P@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$.

where $a, a_i \in \mathcal{A}$ and $\langle\!\langle \tilde{a}\rangle\!\rangle^*_{(\psi,\varphi)}$ is a Constant defined as follows:

$$\begin{aligned}\langle\!\langle a\tilde{a}'\rangle\!\rangle^*_{(\psi,\varphi)} \stackrel{\text{def}}{=}\ & (\textstyle\bigvee_{n\geq 0}[\emptyset]^n\langle a@\psi\rangle\langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\psi,\varphi)}) \\ & \wedge \textstyle\bigwedge\{[b@\varphi]\langle\!\langle a\tilde{a}'\rangle\!\rangle^*_{(\psi,\varphi)} : b \in \mathcal{N}_{ac}\} \\ & \wedge[\emptyset]\langle\!\langle a\tilde{a}'\rangle\!\rangle^*_{(\psi,\varphi)} \\ \langle\!\langle \varepsilon\rangle\!\rangle^*_{(\psi,\varphi)} \equiv\ & \mathtt{tt}\end{aligned}$$

By these sub-lemmas, it is easily shown by induction that if $P@\mathtt{p}|Q@\mathtt{q} \models GR_{(G)}$ and $\tilde{a} \in \mathcal{L}(G)$ then $P@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$. Furthermore, by the definition of $\langle\!\langle \tilde{a}\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$, we can easily obtain that for some P', $P \stackrel{\tilde{a}\,\mathtt{end}}{\Longrightarrow} P'$. This infers that for some Q', $P@\mathtt{p}|Q@\mathtt{q} \stackrel{\{\tilde{a}\,\mathtt{end}@\mathtt{p}\}}{\Longrightarrow} P'@\mathtt{p}|Q'@\mathtt{q}$.

The base case **(4.1.1)** is easier than the other cases. And a proof of **(4.1.3)** is a part of a proof of the case **(4.1.2)**. We show only a proof of **(4.1.2)**.

Proof of (4.1.2) We show that the set $\mathcal{R}$ given in Fig.8 is a satisfaction subset, where we assume that every action name in Fig.8 is contained in $\mathcal{A}_{\mathtt{end}}$. The set $\mathcal{R}$ is divided into 13 subsets $\mathcal{R}_1, \cdots \mathcal{R}_{13}$ with respect to $\tilde{a}_1\tilde{a}\tilde{a}_2$ in $\langle\!\langle \tilde{a}_1\tilde{a}\tilde{a}_2\,\mathtt{end}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ and $(\tilde{a}, \tilde{b}) \in \lambda$ as follows:

- To copy $\tilde{a}_1$ from $\mathtt{p}$ to $\mathtt{q}$: $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$.
  - To read a_1' from $\mathtt{p}$ where $\tilde{a}_1 = a_1'\tilde{a}_1''$: $\mathcal{R}_1$.
  - To wait: $\mathcal{R}_2$.
  - To require a_1' to $\mathtt{q}$: $\mathcal{R}_3$.
- To copy $\tilde{a}$ from $\mathtt{p}$ to $\mathtt{q}$, rewriting according to $(\tilde{a}, \tilde{b}) \in \lambda$, where $\tilde{b} \neq \varepsilon$: $\mathcal{R}_4, \cdots, \mathcal{R}_7$.
  - To repeatedly read $\tilde{a}$ from $\mathtt{p}$: $\mathcal{R}_4$.
  - To wait: $\mathcal{R}_5$.
  - To require $\tilde{b}$ to $\mathtt{q}$: $\mathcal{R}_6, \mathcal{R}_7$.
- To copy $\tilde{a}$ from $\mathtt{p}$ to $\mathtt{q}$, rewriting according to $(\tilde{a}, \varepsilon) \in \lambda$: $\mathcal{R}_8, \mathcal{R}_9$.
  - To repeatedly read $\tilde{a}$ from $\mathtt{p}$: $\mathcal{R}_8$.

$$\mathcal{R} = \bigcup_{1\le i\le 13} \mathcal{R}_i$$

$$\mathcal{R}_1 = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle\!\langle a\tilde{a}_1\tilde{b}_2\tilde{a}_3\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists\tilde{a}_2.\ (\tilde{a}_2,\tilde{b}_2)\in\lambda,$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle a\tilde{a}_1\tilde{a}_2\tilde{a}_3\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle RW_{(G)}\}$$
$$\mathcal{R}_2 = \{(P@\mathtt{p}|Q@\mathtt{q}, [\emptyset]^{n+1}\langle a@\mathtt{q}\rangle\langle\!\langle\tilde{a}_1\tilde{b}_2\tilde{a}_3\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : n\ge 0, \exists\tilde{a}_2.\ (\tilde{a}_2,\tilde{b}_2)\in\lambda,$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset]^n\langle a@\mathtt{p}\rangle\langle\!\langle\tilde{a}_1\tilde{a}_2\tilde{a}_3\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle RW_{(G)}\}$$
$$\mathcal{R}_3 = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle a@\mathtt{q}\rangle\langle\!\langle\tilde{a}_1\tilde{b}_2\tilde{a}_3\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists\tilde{a}_2.\ (\tilde{a}_2,\tilde{b}_2)\in\lambda,$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle\tilde{a}_1\tilde{a}_2\tilde{a}_3\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial\langle a@\mathtt{q}\rangle RW_{(G)}\}$$

$$\mathcal{R}_4 = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists\tilde{a}.\ (\tilde{a},b\tilde{b}')\in\lambda,$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle\tilde{a}\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$
$$\mathcal{R}_5 = \{(P@\mathtt{p}|Q@\mathtt{q}, [\emptyset]^n\langle b@\mathtt{q}\rangle\langle\!\langle\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists\tilde{a}=a_1\cdots a_m.\ m\ge 1, \exists n_{i\ (i\le m)}.\ n=\textstyle\sum_{i\le m}(n_i+1),$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset]^{n_1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$
$$\mathcal{R}_6 = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$
$$\mathcal{R}_7 = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle b@\mathtt{q}\rangle\langle\!\langle\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$

$$\mathcal{R}_8 = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle\!\langle a'\tilde{a}''\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists\tilde{a}.\ (\tilde{a},\varepsilon)\in\lambda, \exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle\tilde{a}a'\tilde{a}''\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$
$$\mathcal{R}_9 = \{(P@\mathtt{p}|Q@\mathtt{q}, [\emptyset]^n\langle a'@\mathtt{q}\rangle\langle\!\langle\tilde{a}''\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) :$$
$$\exists\tilde{a}=a_1\cdots a_m.\ m\ge 1, \exists n'.\ \exists n_{i\ (i\le m)}.\ n=\textstyle\sum_{i\le m}(n_i+1)+n'+1,$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset]^{n_1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle[\emptyset]^{n'}\langle a'@\mathtt{p}\rangle\langle\!\langle\tilde{a}''\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$

$$\mathcal{R}_{10} = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle\!\langle a\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle a\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$
$$\mathcal{R}_{11} = \{(P@\mathtt{p}|Q@\mathtt{q}, [\emptyset]^{n+1}\langle a@\mathtt{q}\rangle\langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : n\ge 0,$$
$$\exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset]^n\langle a@\mathtt{p}\rangle\langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$
$$\mathcal{R}_{12} = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle a@\mathtt{q}\rangle\langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : \exists P_0.\ P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial\langle a@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\}$$

$$\mathcal{R}_{13} = \{(P@\mathtt{p}|Q@\mathtt{q}, \langle\!\langle\varepsilon\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) : P,Q\in Bh\}$$

Fig.8 A satisfaction subset used for proving Sub-lemma (4.1.2)

  - To wait: $\mathcal{R}_9$.
- To copy $\tilde{a}_2$ from $\mathtt{p}$ to $\mathtt{q}$: $\mathcal{R}_{10}, \mathcal{R}_{11}, \mathcal{R}_{12}$.
  - To read a'_2 from $\mathtt{p}$ where $\tilde{a}_2 = a'_2\tilde{a}''_2$: $\mathcal{R}_{10}$.
  - To wait: $\mathcal{R}_{11}$.
  - To require a'_2 to $\mathtt{q}$: $\mathcal{R}_{12}$.
- To terminate: $\mathcal{R}_{13}$.

Then, for each $(C,s)\in\mathcal{R}$, we show that (C,s) satisfies the conditions (i) and (ii) in Definition 3.1 for some $\sigma\in DC(s)$. Here, we show only the cases $(C,s)\in\mathcal{R}_4,\cdots,\mathcal{R}_7$ because the other cases are similar to and easier than these cases.

The case of $(C,s)\in\mathcal{R}_4$: Thus $C\equiv P@\mathtt{p}|Q@\mathtt{q}$, $s\equiv\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $(\tilde{a},b\tilde{b}')\in\lambda$, and $P_0@\mathtt{p}|Q@\mathtt{q}\models\langle\!\langle\tilde{a}\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}\wedge[\tilde{a}@\mathtt{p}]^*\langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Let $\tilde{a}=a_1\cdots a_m$. Since $(\tilde{a},b\tilde{b}')\in\lambda$, $\tilde{a}=a_1\cdots a_m\neq\varepsilon$. Thus $m\ge 1$. By the definition of $\langle\!\langle\tilde{a}\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$, for some $n_{i\ (i\le m)}$, $P_0@\mathtt{p}|Q@\mathtt{q}\models[\emptyset]^{n_1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle\tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$. Now we set σ as follows:

$$\sigma = \{[\emptyset]^n\langle b@\mathtt{q}\rangle\langle\!\langle\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}, [\emptyset]\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}\}$$
$$\cup\{[b''@\mathtt{p}]\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})} : b''\in\mathcal{N}_{ac}\}$$

where $n=\sum_{i\le m}(n_i+1)\ge 1$. Then, we can show that $\sigma\in DC(\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})})$.

Next, the conditions (i) and (ii) are considered.

(i) For any μ and s', $\langle\mu\rangle s' \notin \sigma$.

(ii) Let $[\mu]s' \in \sigma$ and $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\mu} C' \equiv P'@\mathtt{p}|Q'@\mathtt{q}$. There are 5 possible cases.

1. The case; $\mu = \emptyset$, $s' \equiv \langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $n_1 = 0$, and $m = 1$ (i.e. $n = 1$): By the rules **Name** and **Com**, $Q \xrightarrow{1} Q'$ is implied from $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'@\mathtt{p}|Q'@\mathtt{q}$. By the assumption ($P_0@\mathtt{p}|Q@\mathtt{q} \models \langle a_1@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$), for some P'_0 and Q'', $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{a_1@\mathtt{p}\}} P'_0@\mathtt{p}|Q''@\mathtt{q} \models \langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$. This transition implies that $Q \xrightarrow{1} Q''$. Thus, $Q' \equiv Q''$ because the passage of time is determinate. By the other assumption ($P_0@\mathtt{p}|Q@\mathtt{q} \models [a_1@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$), we have that $P'_0@\mathtt{p}|Q'@\mathtt{q} \models \langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ because $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{a_1@\mathtt{p}\}} P'_0@\mathtt{p}|Q''@\mathtt{q}$. Hence, $(C', s') \equiv (P'@\mathtt{p}|Q'@\mathtt{q}, \langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_7$ because $P'_0@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.

2. The case; $\mu = \emptyset$, $s' \equiv [\emptyset]^{n-1}\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $n_1 = 0$, and $m \geq 2$ (i.e. $n \geq 2$): In the same way as the previous case 1, we have that $Q \xrightarrow{1} Q'$ again. Since $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle a_1@\mathtt{p}\rangle[\emptyset]^{n_2}\langle a_2@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$, for some P'_0, $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{a_1@\mathtt{p}\}} P'_0@\mathtt{p}|Q'@\mathtt{q} \models [\emptyset]^{n_2}\langle a_2@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ because the passage of time ($Q \xrightarrow{1} Q'$) is determinate. Furthermore, since $P_0@\mathtt{p}|Q@\mathtt{q} \models \partial[a_1@\mathtt{p}][a_2\cdots a_m@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, we have $P'_0@\mathtt{p}|Q'@\mathtt{q} \models [a_2\cdots a_m@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Here, note that $n-1 = \sum_{2\leq i\leq m}(n_i+1)$ because $m \geq 2$ and $n_1 = 0$. Hence, we obtain $(C', s') \equiv (P'@\mathtt{p}|Q'@\mathtt{q}, [\emptyset]^{n-1}\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_5$ because $m \geq 2$, $n-1 = \sum_{2\leq i\leq m}(n_i+1)$, $P'_0@\mathtt{p}|Q'@\mathtt{q} \models [\emptyset]^{n_2}\langle a_2@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [a_2\cdots a_m@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.

3. The case; $\mu = \emptyset$, $s' \equiv [\emptyset]^{n-1}\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $n_1 \geq 1$, and $m \geq 1$, (i.e. $n \geq 2$): At first, we have that $Q \xrightarrow{1} Q'$ again. Since the passage of time is possible, for some P'_0, $P_0 \xrightarrow{1} P'_0$. These transitions infer $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'_0@\mathtt{p}|Q'@\mathtt{q}$. Thus, by the assumption ($P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset][\emptyset]^{n_1-1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$), we have $P'_0@\mathtt{p}|Q'@\mathtt{q} \models [\emptyset]^{n_1-1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ because $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'_0@\mathtt{p}|Q'@\mathtt{q}$. In addition, by the other assumption ($P_0@\mathtt{p}|Q@\mathtt{q} \models [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$) and the definition of $[\tilde{a}@\mathtt{p}]^*$, we have that $P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset][\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Thus, $P'_0@\mathtt{p}|Q'@\mathtt{q} \models [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ because $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'_0@\mathtt{p}|Q'@\mathtt{q}$. Here, set $n'_{i(i\leq m)}$ as follows: $n'_1 = n_1 - 1$ and $n'_i = n_{i\ (i\geq 2)}$, thus $n - 1 = \sum_{i\leq m}(n'_i + 1)$. Hence, $(C', s') \equiv (P'@\mathtt{p}|Q'@\mathtt{q}, [\emptyset]^{n-1}\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_5$ because $n - 1 = \sum_{i\leq m}(n'_i + 1)$, $P'_0@\mathtt{p}|Q'@\mathtt{q} \models [\emptyset]^{n'_1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n'_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.

4. The case; $\mu = \emptyset$ and $s' \equiv \langle\!\langle b\tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$: In the same way as the previous case 3, for some P'_0, we can show that $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'_0@\mathtt{p}|Q'@\mathtt{q} \models [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Then, since $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$, we have that $P'_0@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ because $\langle\!\langle \tilde{a}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ requires $[\emptyset]\langle\!\langle \tilde{a}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$. Hence, we obtain that $(C', s') \equiv (P'@\mathtt{p}|Q'@\mathtt{q}, \langle\!\langle b\tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_4$ because $(\tilde{a}, b\tilde{b'}) \in \lambda$ and $P'_0@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.

5. The case; $\mu = \{b''@\mathtt{p}\}$ and $s' \equiv \langle\!\langle b\tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$: By **Name** and **Com**, $Q \xrightarrow{1} Q'$ is implied from $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{b''@\mathtt{p}\}} P'@\mathtt{p}|Q'@\mathtt{q}$. In the same way as the previous case 4, we have $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'_0@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Hence, $(C', s') \in \mathcal{R}_4$.

The case of $(C, s) \in \mathcal{R}_5$**:** Thus, $C \equiv P@\mathtt{p}|Q@\mathtt{q}$, $s \equiv [\emptyset]^n\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b'}\tilde{a'}\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $\tilde{a} = a_1\cdots a_m$, $m \geq 1$, $P_0@\mathtt{p}|Q@\mathtt{q} \models [\emptyset]^{n_1}\langle a_1@\mathtt{p}\rangle\cdots[\emptyset]^{n_m}\langle a_m@\mathtt{p}\rangle\langle\!\langle \tilde{a'}\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge [\tilde{a}@\mathtt{p}]^*\langle b\tilde{b'}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, and $n = \sum_{i\leq m}(n_i + 1)$.

Since $n \geq 1$, we set $\sigma = \{[\emptyset]^n\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}\}$, then $\sigma \in DC([\emptyset]^n\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})})$.

Next, the conditions (i) and (ii) are considered.

(i) For any μ and s', $\langle\mu\rangle s' \notin \sigma$.

(ii) Let $[\mu]s' \in \sigma$ and $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\mu} C' \equiv P'@\mathtt{p}|Q'@\mathtt{q}$. Thus, $s' \equiv [\emptyset]^{n-1}\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$ and $\mu = \emptyset$. There are 3 possible cases, but these are the same as the above cases 1, 2, and 3 for $(C, s) \in \mathcal{R}_4$. Hence, we can obtain that $(C', s') \in \mathcal{R}_5 \cup \mathcal{R}_7$.

The case of $(C, s) \in \mathcal{R}_6$**:** Thus, $C \equiv P@\mathtt{p}|Q@\mathtt{q}$, $s \equiv \langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Now we set σ as follows:

$$\sigma = \{\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}, [\emptyset]\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}\} \cup \{[b''@\mathtt{p}]\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})} : b'' \in \mathcal{N}_{ac}\}$$

In this case, $\sigma \in DC(\langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})})$.

Next, the conditions (i) and (ii) are considered.

(i) Let $\langle\mu\rangle s' \in \sigma$. Thus, $\mu = \{b@\mathtt{q}\}$ and $s' \equiv \langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$. Since $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, for some (P_0', Q'), $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{b@\mathtt{q}\}} P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Furthermore, since $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$, we have $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ because $\langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ requires that $[b@\mathtt{q}]\langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$. Here, by **Name** and **Com**, for some P', $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{b@\mathtt{q}\}} P'@\mathtt{p}|Q'@\mathtt{q}$ because $Q \xrightarrow{b} Q'$ and the passage of time ($P \xrightarrow{1} P'$) is always possible. Then, the following three cases are considered.

1. The case; $\tilde{b}' \neq \varepsilon$: $(P'@\mathtt{p}|Q'@\mathtt{q}, \langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_6$ because $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.
2. The case; $\tilde{b}' = \varepsilon$ and for some $a_1\tilde{a}_2 = \tilde{a}'$: Since $P_0'@\mathtt{p}|Q'@\mathtt{q} \models EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, we have $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \partial[a_1@\mathtt{p}]\partial\langle a_1@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ because $a_1 \in \mathcal{A}_{\mathtt{end}}$. Hence, $(P'@\mathtt{p}|Q'@\mathtt{q}, \langle\!\langle a_1\tilde{a}_2\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_{10}$ because $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle a_1\tilde{a}_2\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \partial[a_1@\mathtt{p}]\ \partial\langle a_1@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.
3. The case; $\tilde{b}' = \tilde{a}' = \varepsilon$: $(P'@\mathtt{p}|Q'@\mathtt{q}, \langle\!\langle \varepsilon\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_{13}$.

(ii) Let $[\mu]s' \in \sigma$ and $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\mu} C' \equiv P'@\mathtt{p}|Q'@\mathtt{q}$. There are two possible cases.

1. The case; $\mu = \emptyset$ and $s' \equiv \langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$: By **Name** and **Com**, $Q \xrightarrow{1} Q'$ is implied from $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P'@\mathtt{p}|Q'@\mathtt{q}$. This infers $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P_0'@\mathtt{p}|Q'@\mathtt{q}$ for some P_0' because the passage of time is always possible. Thus, since $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$, we have $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ because $\langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$ requires $[\emptyset]\langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})}$. Also, since $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, we have that $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ because $\langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ requires $[\emptyset]\langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Hence, $(C', s') \equiv (P'@\mathtt{p}|Q'@\mathtt{q}, \langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}) \in \mathcal{R}_6$ because $P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$.
2. The case; $\mu = \{b''@\mathtt{p}\}$ and $s' \equiv \langle\!\langle b\tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$: By **Name** and **Com**, $Q \xrightarrow{1} Q'$ is implied from $P@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{b''@\mathtt{p}\}} P'@\mathtt{p}|Q'@\mathtt{q}$. In the same way as the previous case 1, we have $P_0@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} P_0'@\mathtt{p}|Q'@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Hence, $(C', s') \in \mathcal{R}_6$

The case of $(C, s) \in \mathcal{R}_7$**:** $C \equiv P@\mathtt{p}|Q@\mathtt{q}$, $s \equiv \langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}$, $P_0@\mathtt{p}|Q@\mathtt{q} \models \langle\!\langle \tilde{a}'\rangle\!\rangle^*_{(\mathtt{p},\mathtt{q})} \wedge \langle b\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$. Now set $\sigma = \{\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})}\}$, then $\sigma \in DC(\langle b@\mathtt{q}\rangle\langle\!\langle \tilde{b}'\tilde{a}'\rangle\!\rangle^*_{(\mathtt{q},\mathtt{p})})$. Hence, this case is the same as the case (i) for $(C, s) \in \mathcal{R}_6$. ∎

Proof of Lemma 4.2

Since $\mathbf{CB}_{(G)}$ has no Timeout operator, the transition $\mathbf{CB}_{(G)} \overset{\{\tilde{a}\ \mathtt{end}@\mathtt{p}\}}{\Longrightarrow}$ implies that $\mathbf{CB}_{(G)} \xrightarrow{\{\tilde{a}\ \mathtt{end}@\mathtt{p}\}}$. Thus, $\mathbf{Bh}_{(G)} \xrightarrow{\tilde{a}} \xrightarrow{\mathtt{end}}$. We prove that, for any $n \geq 0$, if $\mathbf{Bh}_{(G,n)} \xrightarrow{\tilde{a}} \xrightarrow{\mathtt{end}}$, then $\tilde{a} \in \bigcup_{i \leq n} \mathcal{L}^{(i)}(G)$, by induction over n.

- The base case ($n = 0$): If $\mathbf{Bh}_{(G,0)} \xrightarrow{\tilde{a}} \xrightarrow{\mathtt{end}} P'$, then $\tilde{a} = a_0 \in \mathcal{L}^{(0)}(G)$.
- The induction case ($n + 1 \geq 1$): Let $\mathbf{Bh}_{(G,n+1)} \xrightarrow{\tilde{a}} \xrightarrow{\mathtt{end}} P'$. By the definition of $\mathbf{Bh}_{(G,n+1)} \equiv \mathbf{Rw}_{(G)}(\mathbf{Bh}_{(G,n)})$, there are two possible cases as follows:
  - For some P'', $\mathbf{Bh}_{(G,n)} \xrightarrow{\tilde{a}} \xrightarrow{\mathtt{end}} P''$ and

$P' \equiv \mathbf{Rw}_{(G)}(P'')$. Hence, by induction, $\tilde{a} \in \bigcup_{i\leq n} \mathcal{L}^{(i)}(G) \subseteq \bigcup_{i\leq n+1} \mathcal{L}^{(i)}(G)$.

– For some $\tilde{a}_1\tilde{a}_2\tilde{a}_3$ and $\tilde{b}_2$, $\tilde{a} = \tilde{a}_1\tilde{b}_2\tilde{a}_3$, $\mathbf{Bh}_{(G,n)} \xrightarrow{\tilde{a}_1}\xrightarrow{\tilde{a}_2}\xrightarrow{\tilde{a}_3}\xrightarrow{\mathtt{end}} P'$, and $(\tilde{a}_2, \tilde{b}_2) \in \lambda$. Hence, by induction, $\tilde{a}_1\tilde{a}_2\tilde{a}_3 \in \bigcup_{i\leq n} \mathcal{L}^{(i)}(G)$. Finally, $\tilde{a} = \tilde{a}_1\tilde{b}_2\tilde{a}_3 \in \bigcup_{i\leq n+1} \mathcal{L}^{(i)}(G)$, because $(\tilde{a}_2, \tilde{b}_2) \in \lambda$.

∎

Proof of Lemma 4.3

We show that the set $\mathcal{R}$ given in Fig.9 is a satisfaction subset, where $\mathcal{A}_{\mathtt{end}} = \mathcal{A} \cup \{\mathtt{end}\}$ and $P \subseteq_+ Q$ means that P is a summand of Q, thus $Q \equiv \cdots + P + \cdots$. Furthermore, Bh^{∂} is the set of behaviors which have neither Timeouts nor Constants. Thus, if $P \in Bh^{\partial}$ and $P \xrightarrow{1} P'$, then $P \equiv P' \in Bh^{\partial}$. For example, $\mathbf{Bh}_{(G)} \in Bh^{\partial}$. Every behavior $P, P', Q, Q', \cdots$, used in this proof is tactically assumed to be contained in Bh^{∂}.

The set $\mathcal{R}$ is divided into 18 subsets $\mathcal{R}_1, \cdots \mathcal{R}_{18}$ as follows:

- To collect all the action-sequence: $\mathcal{R}_1$.
- To idle: $\mathcal{R}_{1,\cdots,5}$.
- To require the initial action: $\mathcal{R}_{6,7}$.
- To copy $(\mathtt{p} \to \mathtt{q})$ with rewriting: $\mathcal{R}_{8,9}$.
- To copy $(\mathtt{p} \to \mathtt{q})$ before rewriting: $\mathcal{R}_{10,\cdots,12}$.
- To copy $(\mathtt{p} \to \mathtt{q})$ after rewriting: $\mathcal{R}_{13,\cdots,15}$.
- To copy $(\mathtt{q} \to \mathtt{p})$: $\mathcal{R}_{16,\cdots,18}$.

Then, for each $(C, s) \in \mathcal{R}$, we show that (C, s) satisfies the conditions (i) and (ii) in Definition 3.1 for some $\sigma \in DC(s)$.

The case of $(C, s) \in \mathcal{R}_1$**:** Thus, $C \equiv \mathbf{CB}_{(G)}$ and $s \equiv GR_{(G)}$. Since $GR_{(G)}$ has no Disjunction, there is an unique σ such that $\sigma \in DC(GR_{(G)})$.

(i) Let $\langle\mu\rangle s' \in \sigma$. By the definition of $GR_{(G)}$, $\mu = \{a_0@\mathtt{p}\}$ and $s' \equiv \partial\langle\mathtt{end}@\mathtt{p}\rangle\mathtt{tt}$. Here, $\mathbf{Bh}_{(G,0)} \xrightarrow{a_0} \mathtt{end}.\mathbf{I}$. Thus, $\mathbf{CB}_{(G)} \xrightarrow{\{a_0@\mathtt{p}\}} \mathtt{end}.\mathbf{I}@\mathtt{p}|\mathbf{Bh}_{(G)}@\mathtt{q}$. Hence, $(\mathtt{end}.\mathbf{I}@\mathtt{p}|\mathbf{Bh}_{(G)}@\mathtt{q}, s') \in \mathcal{R}_6$.

(ii) Let $[\mu]S' \in \sigma$ and $\mathbf{CB}_{(G)} \xrightarrow{\mu} C'$. By the definition of $GR_{(G)}$, there are four possible cases as follows:

1. The case; $\mu = \{a@\mathtt{p}\}$, $s' \equiv [\tilde{a}'@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, and $(a\tilde{a}', \tilde{b}) \in \lambda$: Thus, for some P', $\mathbf{Bh}_{(G)} \xrightarrow{a} P'$ and $C' \equiv P'@\mathtt{p}|\mathbf{Bh}_{(G)}@\mathtt{q}$ because $\mathbf{CB}_{(G)} \xrightarrow{\{a@\mathtt{p}\}} C'$. Furthermore, by the definition of $\mathbf{Bh}_{(G)}$, for some n, $\mathbf{Bh}_{(G,n)} \xrightarrow{a} P'$ and $\mathbf{Rw}_{(G)}(\mathbf{Bh}_{(G,n)}) \equiv \mathbf{Bh}_{(G,n+1)} \subseteq_+ \mathbf{Bh}_{(G)}$. Hence, $(C', s') \in \mathcal{R}_8$.
2. The case; $\mu = \{a@\mathtt{p}\}$, $s' \equiv \partial\langle a@\mathtt{q}\rangle RW_{(G)}$, and $a \in \mathcal{A}_{\mathtt{end}}$: For some P', $\mathbf{Bh}_{(G)} \xrightarrow{a} P'$ and $C' \equiv P'@\mathtt{p}|\mathbf{Bh}_{(G)}@\mathtt{q}$ because $\mathbf{CB}_{(G)} \xrightarrow{\{a@\mathtt{p}\}} C'$. Hence, $(C', s') \in \mathcal{R}_{11}$.
3. The case; $\mu = \{a@\mathtt{q}\}$, $s' \equiv \partial\langle a@\mathtt{p}\rangle EQ^{(\mathtt{q},\mathtt{p})}_{(G)}$, and $a \in \mathcal{A}_{\mathtt{end}}$: For some P', $\mathbf{Bh}_{(G)} \xrightarrow{a} P'$ and $C' \equiv \mathbf{Bh}_{(G)}@\mathtt{p}|P'@\mathtt{q}$, because $\mathbf{CB}_{(G)} \xrightarrow{\{a@\mathtt{q}\}} C'$. Hence, $(C', s') \in \mathcal{R}_{17}$.
4. The case; $\mu = \emptyset$ and $s' \equiv \partial\langle a_0@\mathtt{p}\rangle\partial\langle\mathtt{end}@\mathtt{p}\rangle\mathtt{tt}$ or $s' \equiv [\tilde{a}@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ $((\tilde{a}, \tilde{b}) \in \lambda)$ or $\partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle RW_{(G)}$ or $\partial[a@\mathtt{q}]\partial\langle a@\mathtt{p}\rangle EQ^{(\mathtt{q},\mathtt{p})}_{(G)}$: Since $\mathbf{CB}_{(G)}$ has no timeout, we can infer that $\mathbf{CB}_{(G)} \xrightarrow{\emptyset} C' \equiv \mathbf{CB}_{(G)}$. Hence, $(C', s') \in \bigcup_{2\leq i\leq 5} \mathcal{R}_i$.

The case of $(C, s) \in \mathcal{R}_2 \cup \mathcal{R}_3 \cup \mathcal{R}_4 \cup \mathcal{R}_5$**:** The proof of this case is completely included in the above case $(C, s) \in \mathcal{R}_1$.

The case of $(C, s) \in \mathcal{R}_6$**:** This case is similar to the case (i) in the above case $(C, s) \in \mathcal{R}_1$.

The case of $(C, s) \in \mathcal{R}_7$**:** Trivial.

The case of $(C, s) \in \mathcal{R}_8$**:** $C \equiv P'@\mathtt{p}|Q@\mathtt{q}$, $s \equiv [\tilde{a}'@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, $P \xrightarrow{\tilde{a}} P', \mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ and $(\tilde{a}\tilde{a}', \tilde{b}) \in \lambda$. This case is divided into three sub-cases with respect to the lengths of $\tilde{a}'$ and $\tilde{b}$ as follows:

1. The case; $\tilde{a}' = \varepsilon$ and $\tilde{b} = \varepsilon$: In this case, by the definition of $\mathbf{Rw}_{(G)}(P)$, $P' \subseteq_+$

$$\mathcal{R} = \bigcup_{i \le 18} \mathcal{R}_i$$

$$\mathcal{R}_1 = \{(\mathbf{CB}_{(G)}, GR_{(G)})\}$$
$$\mathcal{R}_2 = \{(\mathbf{CB}_{(G)}, \partial\langle a_0@\mathtt{p}\rangle\partial\langle \mathtt{end}@\mathtt{p}\rangle\mathtt{tt})\}$$
$$\mathcal{R}_3 = \{(\mathbf{CB}_{(G)}, [\tilde{a}@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) : (\tilde{a},\tilde{b}) \in \lambda\}$$
$$\mathcal{R}_4 = \{(\mathbf{CB}_{(G)}, \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle RW_{(G)}) : a \in \mathcal{A}_{\mathtt{end}}\}$$
$$\mathcal{R}_5 = \{(\mathbf{CB}_{(G)}, \partial[a@\mathtt{q}]\partial\langle a@\mathtt{p}\rangle EQ^{(\mathtt{q},\mathtt{p})}_{(G)}) : a \in \mathcal{A}_{\mathtt{end}}\}$$

$$\mathcal{R}_6 = \{(\mathtt{end}.\mathbf{I}@\mathtt{p}|Q@\mathtt{q}, \partial\langle\mathtt{end}@\mathtt{p}\rangle\mathtt{tt}) : Q \in Bh^{\partial}\}$$
$$\mathcal{R}_7 = \{(P@\mathtt{p}|Q@\mathtt{q}, \mathtt{tt}) : P, Q \in Bh^{\partial}\}$$

$$\mathcal{R}_8 = \{(P'@\mathtt{p}|Q@\mathtt{q}, [\tilde{a}'@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) : Q \in Bh^{\partial},$$
$$\exists P \in Bh^{\partial}.\ \mathbf{Rw}_{(G)}(P) \subseteq_+ Q, P \xrightarrow{\tilde{a}} P', (\tilde{a}\tilde{a}', \tilde{b}) \in \lambda\}$$
$$\mathcal{R}_9 = \{(P'@\mathtt{p}|Q@\mathtt{q}, \langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) : Q \in Bh^{\partial}, \exists P \in Bh^{\partial}.\ P \subseteq_+ Q, P \xrightarrow{\tilde{b}} P'\}$$

$$\mathcal{R}_{10} = \{(P@\mathtt{p}|Q@\mathtt{q}, RW_{(G)}) : P, Q \in Bh^{\partial}, \mathbf{Rw}_{(G)}(P) \subseteq_+ Q\}$$
$$\mathcal{R}_{11} = \{(P'@\mathtt{p}|Q@\mathtt{q}, \partial\langle a@\mathtt{q}\rangle RW_{(G)}) : Q \in Bh^{\partial}, \exists P \in Bh^{\partial}.\ \mathbf{Rw}_{(G)}(P) \subseteq_+ Q, P \xrightarrow{a} P'\}$$
$$\mathcal{R}_{12} = \{(P@\mathtt{p}|Q@\mathtt{q}, \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle RW_{(G)}) : P, Q \in Bh^{\partial}, \mathbf{Rw}_{(G)}(P) \subseteq_+ Q\}$$

$$\mathcal{R}_{13} = \{(P@\mathtt{p}|Q@\mathtt{q}, EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) : P, Q \in Bh^{\partial}, P \subseteq_+ Q\}$$
$$\mathcal{R}_{14} = \{(P'@\mathtt{p}|Q@\mathtt{q}, \partial\langle a@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) : Q \in Bh^{\partial}, \exists P \in Bh^{\partial}.\ P \subseteq_+ Q, P \xrightarrow{a} P'\}$$
$$\mathcal{R}_{15} = \{(P@\mathtt{p}|Q@\mathtt{q}, \partial[a@\mathtt{p}]\partial\langle a@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) : P, Q \in Bh^{\partial}, P \subseteq_+ Q\}$$

$$\mathcal{R}_{16} = \{(P@\mathtt{p}|P@\mathtt{q}, EQ^{(\mathtt{q},\mathtt{p})}_{(G)}) : P \in Bh^{\partial}\}$$
$$\mathcal{R}_{17} = \{(P@\mathtt{p}|P'@\mathtt{q}, \partial\langle a@\mathtt{p}\rangle EQ^{(\mathtt{q},\mathtt{p})}_{(G)}) : P \in Bh^{\partial}, P \xrightarrow{a} P'\}$$
$$\mathcal{R}_{18} = \{(P@\mathtt{p}|P@\mathtt{q}, \partial[a@\mathtt{q}]\partial\langle a@\mathtt{p}\rangle EQ^{(\mathtt{q},\mathtt{p})}_{(G)}) : P \in Bh^{\partial}\}$$

Fig.9 A satisfaction subset used for proving Lemma 4.3

$\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ because $P \xrightarrow{\tilde{a}} P'$ and $(\tilde{a}\tilde{a}', \tilde{b}) = (\tilde{a}, \varepsilon) \in \lambda$. And there is an unique σ such that $\sigma \in DC(s) = DC(EQ^{(\mathtt{p},\mathtt{q})}_{(G)})$.

(i) σ has no requirement of the form $\langle\mu\rangle s'$.

(ii) Let $[\mu]s' \in \sigma$ and $C \xrightarrow{\mu} C'$. By the definition of $EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, the following two cases are possible.

(a) The case; $s' \equiv \partial\langle a''@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $\mu = \{a''@\mathtt{p}\}$: Thus, for some P'', $P' \xrightarrow{a''} P''$ and $C' \equiv P''@\mathtt{p}|Q@\mathtt{q}$ because $C \equiv P'@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{a''@\mathtt{p}\}} C'$ and $Q \in Bh^{\partial}$ (i.e. if $Q \xrightarrow{1} Q'$ then $Q \equiv Q'$). Hence, we obtain $(C', s') \in \mathcal{R}_{14}$ because $C' \equiv P''@\mathtt{p}|Q@\mathtt{q}$, $s' \equiv \partial\langle a''@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, $P' \xrightarrow{a''} P''$, and $P' \subseteq_+ Q$.

(b) The case; $s' \equiv \partial[a''@\mathtt{p}]\partial\langle a''@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $\mu = \emptyset$ (idling): Thus, $C \equiv C'$ because $C \equiv P'@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\emptyset} C'$ and $P', Q \in Bh^{\partial}$. Hence, $(C', s') \in \mathcal{R}_{15}$ because $C' \equiv P'@\mathtt{p}|Q@\mathtt{q}$, $s' \equiv \partial[a''@\mathtt{p}]\partial\langle a''@\mathtt{q}\rangle EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $P' \subseteq_+ Q$.

2. The case; $\tilde{a}' = \varepsilon$ and for some $b_1\tilde{b}' = \tilde{b}$: In this case, $\mathbf{Rw}_{(G)}(P) \xrightarrow{b_1} \tilde{b}'.P' \xrightarrow{\tilde{b}'} P'$ because $P \xrightarrow{\tilde{a}} P'$ and $(\tilde{a}\tilde{a}', \tilde{b}) = (\tilde{a}, b_1\tilde{b}') \in \lambda$. Now, we set σ as follows:

$$\sigma = \{\langle b_1@\mathtt{q}\rangle\langle\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)},$$
$$[\emptyset]\langle b_1\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\},$$

then $\sigma \in DC(\langle b_1\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}) = DC(s)$.

(i) Let $\langle\mu\rangle s' \in \sigma$. Thus $s' \equiv \langle\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $\mu = \{b_1@\mathtt{q}\}$. We can infer $C \xrightarrow{\{b_1@\mathtt{q}\}} C' \equiv P'@\mathtt{p}|\tilde{b}'.P'@\mathtt{q}$ from $\mathbf{Rw}_{(G)}(P) \xrightarrow{b_1} \tilde{b}'.P'$ because $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ and $C \equiv P'@\mathtt{p}|Q@\mathtt{q}$. Hence, we obtain $(C', s') \in \mathcal{R}_9$ because $C' \equiv P'@\mathtt{p}|\tilde{b}'.P'@\mathtt{q}$, $s' \equiv \langle\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, and $\tilde{b}'.P' \xrightarrow{\tilde{b}'} P'$.

(ii) Let $[\mu]s' \in \sigma$ and $C \xrightarrow{\mu} C'$. Thus, $s' \equiv \langle b_1\tilde{b}'@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $\mu = \emptyset$. Since $C \xrightarrow{\emptyset} C'$, we have $C' \equiv C \equiv P'@\mathtt{p}|Q@\mathtt{q}$. Hence, we obtain $(C', s') \in \mathcal{R}_9$, because $C' \equiv P'@\mathtt{p}|Q@\mathtt{q}$, $s' \equiv \langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$, and $\mathbf{Rw}_{(G)}(P) \xrightarrow{b} P'$.

3. The case; $\tilde{a}' = a_1\tilde{a}''$ for some $a_1\tilde{a}''$: Now, we set σ as follows:
$$\sigma = \{[a_1@\mathtt{p}][\tilde{a}''@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}, [\emptyset][\tilde{a}'@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}\},$$
then $\sigma \in DC([a_1\tilde{a}''@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)})$.

(i) σ has no requirement of the form $\langle\mu\rangle s'$.

(ii) Let $[\mu]s' \in \sigma$ and $C \xrightarrow{\mu} C'$. There are two possible cases as follows:

(a) The case; $s' \equiv [\tilde{a}''@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $\mu = \{a_1@\mathtt{p}\}$: Thus, for some P'', $P' \xrightarrow{a_1} P''$ and $C' \equiv P''@\mathtt{p}|Q@\mathtt{q}$ because $C \equiv P'@\mathtt{p}|Q@\mathtt{q} \xrightarrow{\{a_1@\mathtt{p}\}} C'$. Hence, $(C', s') \in \mathcal{R}_8$ because $C' \equiv P''@\mathtt{p}|Q@\mathtt{q}$ $s' \equiv [\tilde{a}''@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, $P \xrightarrow{\tilde{a}a_1} P''$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$, and $(\tilde{a}a_1\tilde{a}'', \tilde{b}') \in \lambda$.

(b) The case; $s' \equiv [\tilde{a}'@\mathtt{p}]^*\langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$ and $\mu = \emptyset$: Since $C \xrightarrow{\emptyset} C'$, $C' \equiv C$. Hence, $(C', s') \in \mathcal{R}_8$, because $C' \equiv C$ and $s' \equiv s$.

The case of $(C, s) \in \mathcal{R}_9$**:** $C \equiv P'@\mathtt{p}|Q@\mathtt{q}$, $s \equiv \langle\tilde{b}@\mathtt{q}\rangle^* EQ^{(\mathtt{p},\mathtt{q})}_{(G)}$, $P \subseteq_+ Q$, and $P \xrightarrow{\tilde{b}} P'$. This case is divided into two sub-cases with respect to the length of $\tilde{b}$ as follows:

1. The case; $\tilde{b} = \varepsilon$: In this case, $P \equiv P'$ because $P \xrightarrow{\varepsilon} P'$. This case is exactly the same as the sub-case 1 in the previous case of $(C, s) \in \mathcal{R}_8$ because $P' \subseteq_+ Q$.
2. The case; $\tilde{b} = b_1\tilde{b}'$: In this case, for some P_1, $P \xrightarrow{b_1} P_1 \xrightarrow{\tilde{b}'} P'$ because $P \xrightarrow{\tilde{b}} P'$. Hence, this case is exactly the same as the sub-case 2 in the previous case of $(C, s) \in \mathcal{R}_8$ because $P \xrightarrow{b_1} P_1 \xrightarrow{\tilde{b}'} P'$ and $P \subseteq_+ Q$.

The cases of $(C, s) \in \mathcal{R}_{10,\cdots,18}$**:** These cases are similar to and easier than the cases $\mathcal{R}_{8,9}$. ∎

Proof of Proposition 5.1

Let $\nu_1, \nu_2 \subseteq Act_S$, $\Psi_1, \Psi_2 \subseteq \mathcal{N}_{ag}$, and $\Psi_1 \cap \Psi_2 = \emptyset$. We show that the following $\mathcal{R}$ is a (ν_{12})-im-satisfaction-subset, where $\nu_{12} = (\nu_1\downarrow\Psi_1) \cup (\nu_2\downarrow\Psi_2)$.

$$\mathcal{R} = \{(C_1|C_2, S_1|S_2) : \forall i \in \{1,2\}. \; Agn(C_i) = \Psi_i, C_i \vdash_{\nu_i} S_i\}$$

Let $(C_1|C_2, S_1|S_2) \in \mathcal{R}$.

(i) Let $C_1|C_2 \xrightarrow{\mu} C'$. By **Com**, it implies that for some (μ_1, μ_2, C_1, C_2), $C_1 \xrightarrow{\mu_1} C_1'$, $C_2 \xrightarrow{\mu_2} C_2'$, $\mu = \mu_1 \cup \mu_2$, and $C' = C_1'|C_2'$. For each $i \in \{1,2\}$, since $C_i \vdash_{\nu_i} S_i$, for some S_i', $S_i \overset{\mu_i \cap \nu_i}{\longmapsto} S_i'$ and $C_i' \vdash_{\nu_i} S_i'$. Furthermore, $\mu \cap \nu_{12} = \mu \cap ((\nu_1\downarrow\Psi_1) \cup (\nu_2\downarrow\Psi_2)) = (\mu \cap (\nu_1\downarrow\Psi_1)) \cup (\mu \cap (\nu_2\downarrow\Psi_2)) = (\mu_1 \cap (\nu_1\downarrow\Psi_1)) \cup (\mu_2 \cap (\nu_2\downarrow\Psi_2)) = (\mu_1 \cap \nu_1) \cup (\mu_2 \cap \nu_2)$ because $agn(\mu_i) \subseteq Agn(C_i) = \Psi_i$ and $\Psi_1 \cap \Psi_2 = \emptyset$. Finally, by the definition of the im-Constant $S_1|S_2$, $S_1|S_2 \overset{\mu \cap \nu_{12}}{\longmapsto} S_1'|S_2'$ because $S_1 \overset{\mu_1 \cap \nu_1}{\longmapsto} S_1'$ and $S_2 \overset{\mu_2 \cap \nu_2}{\longmapsto} S_2'$. Furthermore, we have that $(C_1'|C_2', S_1'|S_2') \in \mathcal{R}$ because for each $i \in \{1,2\}$, $agn(C_i') = agn(C_i) = \Psi_i$ and $C_i' \vdash_{\nu_i} S_i'$.

(ii) By a symmetric argument with (i). ∎

Proof of Proposition 5.2

Let $\nu \subseteq Act_S$. We show that the following $\mathcal{R}$ is a (ν)-im-satisfaction-subset.

$$\mathcal{R} = \{(C\backslash M, S\backslash M') : \mathtt{res}(act(C), M, \nu) \subseteq M, \; C \vdash_\nu S, \; M' = \{\mu \cap \nu : \mu \in M\}\}$$

Let $(C\backslash M, S\backslash M') \in \mathcal{R}$.

(i) Let $C\backslash M \xrightarrow{\mu} D'$. By **Res**, it implies that for some C', $C \xrightarrow{\mu} C'$, $\mu \in M \cup \{\emptyset\}$, and $D' = C'\backslash M$. Since $C \vdash_\nu S$, for some S', $S \overset{\mu\cap\nu}{\longmapsto} S'$ and $C' \vdash_\nu S'$. Here, $\mu \cap \nu \in \{\mu' \cap \nu : \mu' \in M \cup \{\emptyset\}\} = M' \cup \{\emptyset\}$ because $\mu \in M \cup \{\emptyset\}$. Hence, by the definition of the im-Constant $S\backslash M'$, $S\backslash M' \overset{\mu\cap\nu}{\longmapsto} S'\backslash M'$ because $S \overset{\mu\cap\nu}{\longmapsto} S'$ and $\mu \cap \nu \in M' \cup \{\emptyset\}$. Furthermore, we have that $(C'\backslash M, S'\backslash M') \in \mathcal{R}$ because $C' \vdash_\nu S'$ and $act(C') \subseteq act(C)$ (i.e. $\mathtt{res}(act(C'), M, \nu) \subseteq \mathtt{res}(act(C), M, \nu) \subseteq M$).

(ii) Let $S \setminus M' \overset{\mu'}{\longmapsto} T'$. By the definition of the im-Constant $S \setminus M'$, for some S', $S \overset{\mu'}{\longmapsto} S'$, $\mu' \in M' \cup \{\emptyset\}$, and $T' \equiv S'\backslash M'$. Since $C \vdash_\nu S$, for some μ_0 and C', $\mu' = \mu_0 \cap \nu$, $C \xrightarrow{\mu_0} C'$ and $C' \vdash_\nu S'$. By the definition of $act(C)$, we have $\mu_0 \in act(C)$ because $C \xrightarrow{\mu_0} C'$.

- The case; $\mu' \neq \emptyset$ and $\mu_0 \neq \emptyset$: Thus $\mu' \in M' = \{\mu \cap \nu : \mu \in M\}$ because $\mu' \in M' \cup \{\emptyset\}$. This means that for some $\mu \in M$, $\mu' = \mu \cap \nu$. Then, $\mu_0 \doteq_\nu \mu$ because $\mu_0 \cap \nu = \mu' = \mu \cap \nu$. Therefore, $\mu_0 \in \mathtt{res}(act(C), M, \nu) \subseteq M$ because $\mu_0 \in act(C)$, $\mu \in M$, and $\emptyset \neq \mu_0 \doteq_\nu \mu$.
- The case; $\mu' = \emptyset$ and $\mu_0 \neq \emptyset$: Thus, $\mu_0 \doteq_\nu \emptyset$ because $\mu_0 \cap \nu = \mu' = \emptyset$. Therefore, $\mu_0 \in \mathtt{res}(act(C), M, \nu) \subseteq M$ because $\mu_0 \in act(C)$, $\emptyset \in M \cup \{\emptyset\}$, and $\emptyset \neq \mu_0 \doteq_\nu \emptyset$.
- The other case; thus $\mu_0 = \emptyset \in M \cup \{\emptyset\}$.

Consequently, always $\mu_0 \in M \cup \{\emptyset\}$. Hence, by **Res**, $C\backslash M \xrightarrow{\mu_0} C'\backslash M$ is inferred from $C \xrightarrow{\mu_0} C'$ and $\mu_0 \in M \cup \{\emptyset\}$. Furthermore, we have that $(C'\backslash M, S'\backslash M') \in \mathcal{R}$ because $C' \vdash_\nu S'$ and $act(C') \subseteq act(C)$. ∎

Proof of Proposition 5.3

Let $\mu, \nu \subseteq Act_S$. We show that the following $\mathcal{R}$ is a $(\nu \cap \mu)$-im-satisfaction-subset.

$$\mathcal{R} = \{(C, S/\mu) : C \vdash_\nu S\}$$

Let $(C, S/\mu) \in \mathcal{R}$, thus $C \vdash_\nu S$.

(i) Let $C \xrightarrow{\mu_1} C'$. Since $C \vdash_\nu S$, for some S', $S \overset{\mu_1\cap\nu}{\longmapsto} S'$ and $C' \vdash_\nu S'$. By the definition of the im-Constant S/μ, it infers that $S/\mu \overset{\mu_1\cap(\nu\cap\mu)}{\longmapsto} S'/\mu$. Here, $(C', S'/\mu) \in \mathcal{R}$.

(ii) By a symmetric argument with (i). ∎

Proof of Proposition 5.4

Only if part (⇒): Let $\nu \subseteq Act_S$. We show that the following $\mathcal{R}$ is a (ν)-satisfaction-subset.

$$\mathcal{R} = \{(S, s) : C \vdash_\nu S, C \models_\nu s\}$$

Let $(S, s) \in \mathcal{R}$, thus $C \vdash_\nu S$ and $C \models_\nu s$. Since $C \models_\nu s$, for some $\sigma \in DC(s)$, $C \models_\nu \bigwedge \sigma$.

(i) Let $\langle\mu\rangle s' \in \sigma$. Since $C \models_\nu \bigwedge \sigma$, for some C' and μ', $C \xrightarrow{\mu'} C'$, $\mu \doteq_\nu \mu'$, and $C' \models_\nu s'$. Also, since $C \vdash_\nu S$, for some S', $S \overset{\mu'\cap\nu}{\longmapsto} S'$ and $C' \vdash_\nu S'$. Hence, $(S', s') \in \mathcal{R}$ because $C' \vdash_\nu S'$ and $C' \models_\nu s'$. Here, $\mu \doteq_\nu \mu' \cap \nu$ because $\mu \doteq_\nu \mu'$ means that $\mu \cap \nu = \mu' \cap \nu = (\mu' \cap \nu) \cap \nu$.

(ii) Let $[\mu]s' \in \sigma$, $S \overset{\mu'}{\longmapsto} S'$, and $\mu \doteq_\nu \mu'$. Since $C \vdash_\nu S$, for some C' and μ'', $\mu' = \mu'' \cap \nu$, $C \xrightarrow{\mu''} C'$, and $C' \vdash_\nu S'$. Here, $\mu \doteq_\nu \mu''$ because $\mu \cap \nu = \mu' \cap \nu = (\mu'' \cap \nu) \cap \nu = \mu'' \cap \nu$. Thus, since $C \models_\nu \bigwedge \sigma$, we have $C' \models_\nu s'$ because $[\mu]s' \in \sigma$, $C \xrightarrow{\mu''} C'$, and $\mu \doteq_\nu \mu''$. Hence, $(S', s') \in \mathcal{R}$.

If part (⇐): Let $\nu \subseteq Act_S$. We can prove that the following $\mathcal{R}$ is a (ν)-satisfaction-subset by a symmetric argument with the "only if part".

$$\mathcal{R} = \{(C, s) : C \vdash_\nu S, S \models_\nu s\}$$

∎

Proof of Proposition 7.1

Only if part (⇒): In this proof, we use a set $Sp^0_\nu(C)$ defined as follows:

$$Sp^0_\nu(C) = \{s \in Sp^0 : C \models_\nu S\}$$

where Sp^0 is defined by removing Constants (recursion) from Sp. In this case, for any $s \in Sp^0$, a negative specification $\neg s$ (i.e. $C \models \neg s$ iff $C \not\models s$) is inductively defined by a well-known way as fol-

lows: $\neg\langle\mu\rangle s \equiv [\mu]\neg s$, $\neg[\mu]s \equiv \langle\mu\rangle\neg s$, $\neg\bigwedge_{i\in I} s_i \equiv \bigvee_{i\in I}\neg s_i$, and $\neg\bigvee_{i\in I} s_i \equiv \bigwedge_{i\in I}\neg s_i$.

We show that the following $\mathcal{R}$ is a (ν)-im-satisfaction-subset because $Sp_\nu(C) = Sp_\nu(D)$ implies $Sp^0_\nu(C) = Sp^0_\nu(D)$.

$\mathcal{R} = \{(C,S) : \exists D.\ Sp^0_\nu(C) = Sp^0_\nu(D), D \vdash_\nu S\}$

We use a contradiction, thus assume that for some $(C,S) \in \mathcal{R}$, either

($\neg i$) for some μ and C', $C \xrightarrow{\mu} C'$, for every S' such that $S \overset{\mu\cap\nu}{\longmapsto} S'$, $(C',S') \notin \mathcal{R}$, or

($\neg ii$) for some μ and S', $S \overset{\mu}{\longmapsto} S'$, for every C' and μ' such that $C \xrightarrow{\mu'} C'$ and $\mu = \mu' \cap \nu$, $(C',S') \notin \mathcal{R}$.

Since $(C,S) \in \mathcal{R}$, for some D, $Sp^0_\nu(C) = Sp^0_\nu(D)$ and $D \vdash_\nu S$. First, assume $(\neg i)$, thus for some μ and C', $C \xrightarrow{\mu} C'$. And set μ_i and D_i $(i \in I)$ as follows:

$\{(\mu_i, D_i) : i \in I\} = \{(\mu', D') : D \xrightarrow{\mu'} D', \mu \doteq_\nu \mu'\}$

Then, for each $i \in I$, for some s_i, $C' \models s_i$ and $D_i \not\models s_i$ as follows:

> Since $D \vdash_\nu S$, for some S_i, $S \overset{\mu_i\cap\nu}{\longmapsto} S_i$ and $D_i \vdash_\nu S_i$ because $D \xrightarrow{\mu_i} D_i$. Here, $\mu_i \cap \nu = \mu \cap \nu$ because $\mu \doteq_\nu \mu_i$. Thus, by $(\neg i)$, $(C',S_i) \notin \mathcal{R}$. Furthermore, by the definition of $\mathcal{R}$, $Sp^0_\nu(C') \neq Sp^0_\nu(D_i)$ because $D_i \vdash_\nu S_i$. Hence, for some $s_i \in Sp^0$, $C' \models s_i$ and $D_i \not\models s_i$.

Now set $s \equiv \langle\mu\rangle(\bigwedge_{i\in I} s_i) \in Sp^0$. Then, $C \models s$ because $C \xrightarrow{\mu} C'$ and for every $i \in I$, $C' \models s_i$. On the other hand, $D \not\models s$ because for any transition such that $D \xrightarrow{\mu'} D'$ and $\mu \doteq_\nu \mu'$, for some $i \in I$, $D' \equiv D_i \not\models s_i$. But, they ($C \models s$ and $D \not\models s$) contradict the assumption $Sp^0_\nu(C) = Sp^0_\nu(D)$. Hence $(\neg i)$ is impossible.

The second assumption $(\neg ii)$ is also impossible by a symmetric argument with the case of $(\neg i)$. Consequently $\mathcal{R}$ is a (ν)-im-satisfaction-subset.

If part ($\Leftarrow$): Assume that $ISp_\nu(C) = ISp_\nu(D)$. Here, $ISp_\nu(C) \neq \emptyset$ because $C \vdash_\nu \mathtt{imsp}_\nu(C)$, where $\mathtt{imsp}_\nu(C)$ is defined in Section 7. Thus, for some S, $C \vdash_\nu S$ and $D \vdash_\nu S$.

We show that the following set $\mathcal{R}$ is a (ν)-satisfaction-subset.

$\mathcal{R} = \{(C,s) : \exists S.\ C \vdash_\nu S, D \vdash_\nu S, D \models_\nu s\}$

Let $(C,s) \in \mathcal{R}$, thus for some S, $C \vdash_\nu S$, $D \vdash_\nu S$, and $D \models_\nu s$. Since $D \models_\nu s$, for some $\sigma \in DC(s)$, $D \models_\nu \bigwedge\sigma$.

(i) Let $\langle\mu\rangle s' \in \sigma$. Since $D \models_\nu \bigwedge\sigma$, for some D' and μ', $D \overset{\mu'}{\longmapsto} D'$, $\mu \doteq_\nu \mu'$, and $D' \models_\nu s'$. Also, since $D \vdash_\nu S$, for some S', $S \overset{\mu'\cap\nu}{\longrightarrow} S'$ and $D' \vdash_\nu S'$. Furthermore, since $C \vdash_\nu S$, for some C' and μ'', $\mu' \cap \nu = \mu'' \cap \nu$, $C \xrightarrow{\mu''} C'$, and $C' \vdash_\nu S'$. Hence, $(C',s') \in \mathcal{R}$ because $C' \vdash_\nu S'$, $D' \vdash_\nu S'$, and $D' \models_\nu s'$. And, $\mu \doteq_\nu \mu' \doteq_\nu \mu''$.

(ii) Let $[\mu]s' \in \sigma$, $C \xrightarrow{\mu'} C'$, and $\mu \doteq_\nu \mu'$. Then, we can show that $(C',s') \in \mathcal{R}$ by a symmetric argument with the case of (i). ∎