The Decidability of the Reachability Problem for CCS$^!$ *

Chaodong He

BASICS, Department of Computer Science
Shanghai Jiao Tong University, Shanghai 200240, China
MOE-MS Key Laboratory for Intelligent Computing and Intelligent Systems

Abstract. CCS$^!$ is a variant of CCS in which infinite behaviours are defined by the replication operator. We show that the reachability problem for CCS$^!$ is decidable by a reduction to the same problem for Petri Nets.

1 Introduction

Process calculi provide languages in which the structure of syntactic terms represents the structure of processes and the operational semantics represents steps of computation or interaction. Among various process calculi, CCS remains a standard representative.

Several variants of CCS have appeared in literature. The relative expressive power of these variants is investigated in [5,6,7,11,10]. It seems that there are two aspects which affect the expressive power significantly. One is the mechanism adopted for extending finite processes in order to express infinite behaviours [5]. The other is the capability of producing and manipulating local channels [13]. According to this fact, five major variants of CCS are given in Fig. 1. In the diagram an arrow '$\longrightarrow$' indicates the sub-language relationship. The five variants of CCS are further divided into three classes. The first class contains CCS$^{\mathrm{Pdef}}$, in which infinite behaviours are specified by *parametric definition* [22,11] (or equivalently *dynamic-scoping recursion* [5]). This mechanism offers a certain degree of name-passing capability such that process copies can be nested at arbitrary depth, which results in the Turing completeness of CCS$^{\mathrm{Pdef}}$ [11,5,25]. The second class contains CCS$^{\mu}$ and CCS$^!$, in which the infinite behaviours are specified by (static-scoping) recursion and replication, respectively. These two subcalculi have the power of producing new local channels but not have the power of passing names around. They are not Turing complete because they are not expressive enough to define 'counter' [10]. The third class contains CCS$^{\mu}_{\bullet}$ and CCS$^!_{\bullet}$ in which the local names are always static. In these two variants, localization operators can only act as the outermost constructors to ensure that no local channels can be produced during the evolution of processes.

Given a variant of CCS, a legitimate question is whether a certain process property is decidable. This paper explores the reachability problem for the CCS

* The work is supported by NSFC (60873034, 61033002).

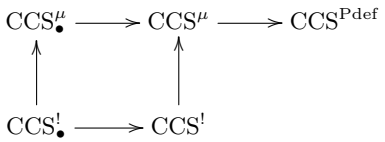

Fig. 1. CCS Variants

variants. This problem asks whether a given source process can evolve to a given target process within finitely many steps of computation or interaction.

The reachability problem for $\mathrm{CCS}^{\mathrm{Pdef}}$ is undecidable due to Turing completeness. The reachability problem for $\mathrm{CCS}^{\mu}_{\bullet}$ and $\mathrm{CCS}^{!}_{\bullet}$ is decidable, which is obtained from the fact that $\mathrm{CCS}^{\mu}_{\bullet}$ and $\mathrm{CCS}^{!}_{\bullet}$ can be embedded into Labelled Petri Nets (LPN for short) with simple modification of U.Goltz's encoding [12] by C.He *et al.* [13], and from a prominent discovery for Petri Nets (or equivalently *Vector Addition System*), that *the reachability problem for Petri Nets is decidable*. This central problem is known decidable by algorithms based on the classical Kosaraju-Lambert-Mayr-Sacerdote-Tenny decomposition (KLMTS decomposition) [24,19,15,18].

The contribution of this paper is to show that the reachability problem for $\mathrm{CCS}^{!}$ is decidable. The result is proved by a reduction to the reachability problem for Labelled Petri Nets.

At first we notice that the way of deciding reachability problem for $\mathrm{CCS}^{!}_{\bullet}$ (or $\mathrm{CCS}^{\mu}_{\bullet}$) does not work for $\mathrm{CCS}^{!}$. The standard encodings from CCS to LPN [25,12] share the guideline that the sequential processes are represented by places and their parallel occurrences are counted by tokens. These encodings do nothing with local names. This is why the encoding from $\mathrm{CCS}^{!}_{\bullet}$ (or $\mathrm{CCS}^{\mu}_{\bullet}$) to LPN relies heavily on static local names. Intuitively, in $\mathrm{CCS}^{!}$ there are processes, for instance of the form $!(\ldots \parallel (a)!P \parallel (b)!Q \parallel \ldots)$, in which nested local names form a tree. The standard encodings may cause tokens for different component confused. There are other evidences which suggest that no reasonable encoding from $\mathrm{CCS}^{!}$ to LPN exists. In [6], N.Busi *et al.* show that $\mathrm{CCS}^{!}$ can model Minsky Machine non-deterministically, which confirms that $\mathrm{CCS}^{!}$ is 'nearly' Turing complete, while such a result seems not to hold for LPN, which is likely to be 'far from' Turing complete. A more convincing fact is that, for $\mathrm{CCS}^{!}$ strong bisimilarity with a given regular process is undecidable [13], while this problem is decidable for LPN [14]. Even though CCS can indeed be encoded in terms of LPN with infinite places or with inhibitor arcs [8], it is helpless in deciding reachability problem for CCS, for these accessories make the corresponding reachability problem for Petri Nets undecidable.

The key observation yielding the decidability result is the following property of $\mathrm{CCS}^{!}$: When a replicated subprocess becomes 'active' (i.e. not guarded by a prefix), this subprocess remains active evermore. Based on this observation, if there exists an evolution path from source to target, the number of active occurrences of a certain replicated subprocess in any intermediate states is 'bounded'

$$\textsf{Choice}\ \frac{}{\sum_{i=1}^{n}\lambda_i.P_i \xrightarrow{\lambda_i} P_i} \qquad \textsf{Composition}\ \frac{P \xrightarrow{\lambda} P'}{P \parallel Q \xrightarrow{\lambda} P' \parallel Q} \quad \frac{P \xrightarrow{l} P' \quad Q \xrightarrow{\bar{l}} Q'}{P \parallel Q \xrightarrow{\tau} P' \parallel Q'}$$

$$\textsf{Localization}\ \frac{P \xrightarrow{\lambda} P' \quad a \text{ not appear in } \lambda}{(a)P \xrightarrow{\lambda} (a)P'} \qquad \textsf{Replication}\ \frac{P \xrightarrow{\lambda} P'}{!\,P \xrightarrow{\lambda} !\,P \parallel P'}$$

Fig. 2. Semantics of CCS$^!$

by that number in the target. Within these 'bounds', predefined in the target process, a Labelled Petri Net is constructed recursively. These 'bounds' serve as the requisite copies of subnets for representing every replicated subprocess. The constructed net is strong enough to produce the evolution path in which the numbers of active replicated subprocesses are 'bounded'.

The rest of the paper is organized as follows. Section 2 lays down the preliminaries. Section 3 expounds the main idea and formal definitions. Section 4 describes the construction of Labelled Petri Net. Section 5 states the whole algorithm. Section 6 gives an illustrating example. Section 7 concludes.

2 Preliminaries

2.1 The Calculus

To describe the interactions between systems, we need channel names. The set of the names $\mathcal{N}$ is ranged over by $a, b, c, \ldots$, and the set of the names and the conames $\mathcal{N} \cup \overline{\mathcal{N}}$ is ranged over by $l, \ldots$. The set of the action labels $\mathcal{A} = \mathcal{N} \cup \overline{\mathcal{N}} \cup \{\tau\}$ is ranged over by λ.

The set $\mathcal{P}_{\text{CCS}^!}$ of CCS$^!$ processes, ranged over by $P, Q, \ldots$, is generated inductively by the grammar

$$P \quad ::= \quad \mathbf{0} \quad | \quad \sum_{i=1}^{n} \lambda_i.P_i \quad | \quad P \parallel P' \quad | \quad (a)P \quad | \quad !\,P$$

A name a appeared in process $(a)P$ is *local*. A name is *global* if it is not local. We write $\mathsf{gn}(P)$ for the set of global names of P.

The semantics of CCS$^!$ is given by *labelled transition system* $(\mathcal{P}_{\text{CCS}^!}, \mathcal{A}, \longrightarrow)$, where the elements of $\mathcal{P}_{\text{CCS}^!}$ are often referred to as *states*. The relation $\longrightarrow \subseteq \mathcal{P}_{\text{CCS}^!} \times \mathcal{A} \times \mathcal{P}_{\text{CCS}^!}$ is the *transition* relation. The membership $(P, \lambda, P') \in \longrightarrow$ is always indicated by $P \xrightarrow{\lambda} P'$. The relation $\longrightarrow$ is generated inductively by the rules defined in Fig. 2. The symmetric rules are omitted.

Standard notations and conventions in process calculi will be used throughout the paper. The inactive process $\mathbf{0}$ is omitted in most occasions. For instance $a.b.\mathbf{0}$ is abbreviated to $a.b$. A finite sequence (or set) of names $a_1, \ldots, a_n$ is

often abbreviated to $\widetilde{a}$. The guarded choice term $\sum_{i=1}^{n} \lambda_i.P_i$ is usually written as $\lambda_1.P_1 + \cdots + \lambda_n.P_n$. Processes are not distinguished syntactically up to the commutative monoid generated by '+' and '$\|$'. We shall write $\prod_{i=1}^{n} P_i$ for $P_1 \| \cdots \| P_n$. The notation '$\equiv$' is used to indicate syntactic congruence. The set of the *derivatives* of a process P, denoted by $\mathtt{Drv}(P)$, is the set of the processes P' such that $P \xrightarrow{\lambda_1} \cdots \xrightarrow{\lambda_n} P'$ for some $n \geq 0$ and $\lambda_1, \ldots, \lambda_n \in \mathcal{A}$.

2.2 The Petri Nets

Let $\mathbb{N}$ be the set of natural numbers. A *Petri Net* is a tuple $N = (S, T, F, \mathbf{m}_{\text{init}})$ and a *Labelled Petri Net* is a tuple $N = (S, T, F, L, \mathbf{m}_{\text{init}})$, where S and T are finite disjoint sets of *places* and *transitions* respectively, $F : (S \times T) \cup (T \times S) \to \mathbb{N}$ is a *flow function* and $L : T \to \mathcal{A}$ is a *labelling*. $\mathbf{m}_{\text{init}}$ is the *initial marking*, where a *marking* $\mathbf{m}$ is a function $S \to \mathbb{N}$ assigning the number of *tokens* to each place.

A transition $t \in T$ is *enabled* at a marking $\mathbf{m}$, denoted by $\mathbf{m} \overset{t}{\rightsquigarrow}$, if $\mathbf{m}(\mathtt{s}) \geq F(\mathtt{s}, t)$ for every $\mathtt{s} \in S$. A transition t enabled at $\mathbf{m}$ may *fire* yielding the marking $\mathbf{m}'$, denoted by $\mathbf{m} \overset{t}{\rightsquigarrow} \mathbf{m}'$, where $\mathbf{m}'(\mathtt{s}) = \mathbf{m}(\mathtt{s}) - F(\mathtt{s}, t) + F(t, \mathtt{s})$ for all $\mathtt{s} \in S$. For each $\lambda \in \mathcal{A}$, we write $\mathbf{m} \overset{\lambda}{\rightsquigarrow}$, respectively $\mathbf{m} \overset{\lambda}{\rightsquigarrow} \mathbf{m}'$ to mean that $\mathbf{m} \overset{t}{\rightsquigarrow}$, respectively $\mathbf{m} \overset{t}{\rightsquigarrow} \mathbf{m}'$ for some t with $L(t) = \lambda$. A labelled transition system $(\mathcal{M}, \mathcal{A}, \rightsquigarrow)$ can be generated from a Labelled Petri Net N, where $\mathcal{M}$ is the set of all markings of N.

In the remainder of this paper, Labelled Petri Nets are treated more algebraically. Let $S = \{\mathtt{s}_i\}_{i=1}^{|S|}$ be the set of places of N. A marking $\mathbf{m} = \{m_i\}_{i=1}^{|S|}$ is viewed as a vector with dimension $|S|$, or equivalently a multiset over S. A transition t will be specified by a label λ and two vectors $\mathbf{v} = \{v_i\}_{i=1}^{|S|}$ and $\mathbf{w} = \{w_j\}_{j=1}^{|S|}$. The flow function F for t is defined by $F(\mathtt{s}_i, t) = v_i$ and $F(t, \mathtt{s}_j) = w_j$ for every $i, j \in \{1, \ldots, |S|\}$. We will use *labelled transition rules* of the form

$$\mathtt{s}_{i_1}^{v_{i_1}} \mathtt{s}_{i_2}^{v_{i_2}} \ldots \mathtt{s}_{i_p}^{v_{i_p}} \overset{\lambda}{\rightsquigarrow} \mathtt{s}_{j_1}^{w_{i_1}} \mathtt{s}_{j_2}^{w_{i_2}} \ldots \mathtt{s}_{j_q}^{w_{i_q}}$$

to indicate a transition t with label λ, vectors $\{v_i\}_{i=1}^{|S|}$ and $\{w_j\}_{j=1}^{|S|}$, where v_i and w_j is zero if $i \notin \{i_1, \ldots, i_p\}$ or $j \notin \{j_1, \ldots, j_q\}$. Whenever $\mathbf{m} = \mathbf{r} + \mathbf{v}$, $\mathbf{m}$ can be replaced by $\mathbf{m}' = \mathbf{r} + \mathbf{w}$. The empty multiset is denoted by ϵ. Thus an Labelled Petri Net N is specified by $(S, \mathcal{A}, \rightsquigarrow, \mathbf{m}_{\text{init}})$, where $\rightsquigarrow \in \mathcal{M} \times \mathcal{A} \times \mathcal{M}$ is a set of labelled transition rules.

2.3 Reachability Problem

The formalization of the reachability problem depends on when two processes are regarded syntactically equal. Let $\doteq$ be an equivalence relation on $\mathcal{P}_{\text{CCS}}$. We have the following parameterized reachability problem:

Problem:	REACHABILITY(CCS, $\doteq$)
Instance:	Two CCS processes P and Q.
Question:	Does there exist Q' such that $Q \doteq Q' \in \mathtt{Drv}(P)$?

The relation $\backsimeq$ serves as the syntactical equality. The question here is how shall we choose $\backsimeq$? The syntactic nature requires that $\backsimeq$ must be decidable, and it also should validate that following *harmonic property*:

If $P_1' \backsimeq P_1 \xrightarrow{\lambda} P_2$, then there exists P_2' such that $P_1' \xrightarrow{\lambda} P_2' \backsimeq P_2$.

The harmonic property is exactly the strong bisimulation property [21,23]. We require that the inference of $P_1' \xrightarrow{\lambda} P_2'$ can be effectively constructed.

The strong bisimilarity itself is not a good candidate of $\backsimeq$. Using the construction of Busi, Gabbrielli, and Zavattaro [6], one can show Reachability(CCS$^!$, $\sim$) undecidable. On the other hand, Reachability(CCS$^!$, $\backsimeq$) could be decided in an obvious way if we not imposing requisite equations on $\backsimeq$. For example if P is not equated to $P \parallel \mathbf{0}$, Reachability(CCS$^!$, $\backsimeq$) can be decided with the intuition that, during the evolution, the number of unguarded composition operators (not appear under guarded choice) cannot decrease, and every infinite evolution path must eventually using the rule for replication, which increases this number strictly.

Definition 1. *The* strong structural congruence*, $\equiv$, is the smallest congruence relation generated by the following laws:*

$$P \parallel Q \equiv Q \parallel P \qquad (P \parallel Q) \parallel R \equiv P \parallel (Q \parallel R) \qquad P \parallel \mathbf{0} \equiv P$$

The weak structural congruence*, $\dot{\equiv}$, is the smallest congruence generated by the laws for $\equiv$ together with the following laws:*

$$(a_1)(a_2)P \dot{\equiv} (a_2)(a_1)P \qquad (a)(P \parallel Q) \dot{\equiv} P \parallel (a)Q \;\; \textit{if } a \notin \mathsf{gn}(P) \qquad (a)\mathbf{0} \dot{\equiv} \mathbf{0}$$

In this paper, we treat $\backsimeq$ to be $\equiv$ or $\dot{\equiv}$. The reachability problem for CCS$^!$ always refers to Reachability(CCS$^!$, $\equiv$) or Reachability(CCS$^!$, $\dot{\equiv}$).

3 Main Idea

3.1 Informal Description

It is mentioned in Section 1 that the reachability problem for $\mathrm{CCS}^!_\bullet$ can be decided by a structural encoding to LPN. Each process $P \in \mathcal{P}_{\mathrm{CCS}^!_\bullet}$ can be assumed to be in the form $(\tilde{a}) \prod_{i \in I} P_i$ in which $\tilde{a}$ are all the local names of P, and every P_i, named *concurrent component*, is localization free and is not a composition. The encoding depends on the fact that local names are static, and the number of the possible concurrent components of all derivatives of P is finite. The encoding works for $\mathrm{CCS}^\mu_\bullet$ as well [13]. However, this encoding is not sound if local names can appear underneath replicators. In these situations, the local names newly produced may be capable of preventing certain interactions between components, which is unknown before running the process.

The basic idea of our deciding algorithm for the reachability problem of CCS$^!$ is motivated by the following observation. Consider the following two processes

$$\begin{aligned} P &\stackrel{\text{def}}{=} !c.!((a.!P_1 + b.!P_2) \parallel !P_3) \\ Q &\stackrel{\text{def}}{=} !c.!((a.!P_1 + b.!P_2) \parallel !P_3) \parallel \\ &\quad !((a.!P_1 + b.!P_2) \parallel !P_3) \parallel !P_1 \parallel !P_3 \parallel \\ &\quad !((a.!P_1 + b.!P_2) \parallel !P_3) \parallel !P_2 \parallel P_2' \parallel !P_3 \end{aligned}$$

where $P_2 \xrightarrow{\lambda} P_2'$. It is easy to check that $Q \equiv \in \mathtt{Drv}(P)$. We find in the definition of Q that the number of active (i.e. not guarded by a prefix) occurrences of the replicated process $!P_1$, $!P_2$, and $!P_3$ are *one*, *one*, and *two*, respectively. The key observation is that, once a replicated subprocess becomes active, this subprocess remains always active. Based on this observation, any intermediate states in every evolution path from P to Q must have at most one active occurrence of $!P_1$, one active occurrence of $!P_2$, and two active occurrences of $!P_3$.

The main difficulty in translating CCS$^!$ to LPN is that the interplay of localization and replication have the power of creating unbounded number of different components (modulo $\equiv$ or $\dot{\equiv}$). Because each component is usually interpreted as a place, it seems that infinite places are needed. The above observation enlightens us that some computation paths can be excluded, and in the remaining computation paths, at most finite number of different components can emerge. Within this insight, a Labelled Petri Net is constructed recursively, in which every component is translated into a certain copy of a sub-net.

3.2 Formal Definitions

In order to formalize the above intuition, we need several auxiliary notations.

Let $P \in \mathcal{P}_{\text{CCS}^!}$, and $P' \in \mathtt{Drv}(P)$. A component P_i of P' may be created during the evolution by applying rule Replication. In this situation, it is helpful for us to know the place P_i comes from. This suggests the following.

Definition 2. *Let $\mathcal{T}$ be the set of tags, ranged over by $\mathsf{u}, \mathsf{v}, \ldots$. The processes of* CCS$^!$ *with tags is generated inductively by the grammar*

$$P \quad ::= \quad \mathbf{0} \quad | \quad \sum_{i=1}^{n} \lambda_i.P_i \quad | \quad P \parallel P' \quad | \quad (a)P \quad | \quad !P \quad | \quad \langle P \rangle_{\mathsf{v}}$$

The process $\langle P \rangle_{\mathsf{v}}$ is in tagged form. *The semantic rule for tag is*

$$\mathsf{Tag} \ \frac{P \xrightarrow{\lambda} P'}{\langle P \rangle_{\mathsf{v}} \xrightarrow{\lambda} \langle P' \rangle_{\mathsf{v}}}$$

The (weak) structural congruence is generated to tagged processes by letting $\langle \mathbf{0} \rangle_{\mathsf{v}} \equiv \mathbf{0}$. By congruence we have $\langle (a)\mathbf{0} \rangle_{\mathsf{v}} \dot{\equiv} \mathbf{0}$ and $(a)\langle \mathbf{0} \rangle_{\mathsf{v}} \dot{\equiv} \mathbf{0}$. Note that we do not have equations such as $\langle P \parallel Q \rangle_{\mathsf{v}} \equiv P \parallel \langle Q \rangle_{\mathsf{v}}$ in general.

Definition 3. *A process P of* CCS$^!$ *is* standard*, if the subprocesses of P in tagged form are exactly the ones just underneath replication operators, and every tag in P is different.*

For example, neither $!\langle a + b.!\,c.d\rangle_{\mathsf{v}}$ nor $!\langle a\rangle_{\mathsf{v}} \parallel !\langle a\rangle_{\mathsf{v}}$ is standard, because in the former $c.d$ is not in tagged form, and in the latter the same tag v appears twice. These processes can be rectified to $!\langle a + b.!\langle c.d\rangle_{\mathsf{v}_2}\rangle_{\mathsf{v}_1}$ and $!\langle a\rangle_{\mathsf{v}_1} \parallel !\langle a\rangle_{\mathsf{v}_2}$, which is now standard.

The next notation is used to specify standard processes or their derivatives.

Definition 4. *The set $\mathcal{C}$ of* context*, ranged over by $\mathbf{C}$, is generated inductively by the grammar*

$$\mathbf{C} \quad ::= \quad \underline{\mathsf{v}} \quad | \quad \mathbf{0} \quad | \quad \sum_{i=1}^{n} \lambda_i.\mathbf{C}_i \quad | \quad \mathbf{C} \parallel \mathbf{C}' \quad | \quad (a)\mathbf{C} \quad | \quad \langle \mathbf{C}\rangle_{\mathsf{v}}$$

We use $\mathbf{C}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ to indicate a context with tags exactly $\mathsf{v}_1, \mathsf{v}_2, \ldots, \mathsf{v}_n$. $\mathbf{C}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ are often abbreviated as $\mathbf{C}$ if no ambiguity arises. We use $\mathbf{C}[P_1, P_2, \ldots, P_n]$ to indicate the process by replacing each $\underline{\mathsf{v}_i}$ with $!\langle P_i\rangle_{\mathsf{v}_i}$. We use $\mathbf{C}\{R/\underline{\mathsf{v}_i}\}$ to indicate the process by replacing the hole $\underline{\mathsf{v}_i}$ with the process R.

A standard process P can be represented as:

$$\mathbf{C}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}, \ldots, P_{\mathsf{v}_n}]$$

in which every v_i occurs only once and every P_{v_i} is standard for $i = 1, \ldots, n$. We will call $\mathbf{C}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ the *characteristic context* of P. The derivatives of P can also be represented in this way where the same v_i can occur more than once. For example, let P be the process $b.!\langle c.(a)(!\langle a\rangle_{\mathsf{v}_1} \parallel d.!\langle \overline{a}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}}$. P can be represented as $\mathbf{C}[P_{\mathsf{u}}]$ with $\mathbf{C}[\underline{\mathsf{u}}] = b.\underline{\mathsf{u}}$, and P_{u} is represented as $\mathbf{C}_{\mathsf{u}}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}]$ with $\mathbf{C}_{\mathsf{u}}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}] = c.(a)(\underline{\mathsf{v}_1} \parallel d.\underline{\mathsf{v}_2})$, $P_{\mathsf{v}_1} \equiv a$ and $P_{\mathsf{v}_2} \equiv \overline{a}$. When

$$P \xrightarrow{b} \xrightarrow{c} P' \equiv !\langle c.(a)(!\langle a\rangle_{\mathsf{v}_1} \parallel d.!\langle \overline{a}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel \langle (a)(!\langle a\rangle_{\mathsf{v}_1} \parallel d.!\langle \overline{a}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}},$$

P' can be represented as $\mathbf{C}'[P_{\mathsf{u}}, P_{\mathsf{v}_1}, P_{\mathsf{v}_2}]$ with $\mathbf{C}'[\underline{\mathsf{u}}, \underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}] = \underline{\mathsf{u}} \parallel \langle (a)(\underline{\mathsf{v}_1} \parallel d.\underline{\mathsf{v}_2})\rangle_{\mathsf{u}}$.

Definition 5. *Let $\mathbf{C}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ be a context with tags $\mathsf{v}_1, \mathsf{v}_2, \ldots, \mathsf{v}_n$. We say that v_i is* active *in $\mathbf{C}$, denoted by $\mathbf{C} \rhd \mathsf{v}_i$, if v_i is not under guarded choice. The set of active tags in $\mathbf{C}$ is denoted by $\mathtt{Act}(\mathbf{C})$.*

Let P be standard and $P' \in \mathtt{Drv}(P)$. When $P' = \mathbf{C}'[P_{\mathsf{v}_1}, \ldots, P_{\mathsf{v}_n}]$ and $\mathbf{C}' \rhd \mathsf{v}_i$, we say that $!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}$ is active *in P', denoted by $P' \rhd !\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}$.*

The number of active occurrences of v_i in $\mathbf{C}'$ (or $!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}$ in P') is denoted by $\mathbf{num}(\mathsf{v}_i, \mathbf{C}')$ (or $\mathbf{num}(!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}, P')$).

For example, $!\langle b.c\rangle_{\mathsf{v}}$ is active in $(b)(!\langle b.c\rangle_{\mathsf{v}} \parallel d)$, while $!\langle b.c\rangle_{\mathsf{v}}$ is not active in $(b)d.(!\langle b.c\rangle_{\mathsf{v}} \parallel e)$. Note also that it makes no sense to talk about whether $!\langle b.c\rangle_{\mathsf{v}}$ is active in $!\langle !\langle b.c\rangle_{\mathsf{v}}\rangle_{\mathsf{u}}$, for v does not occur in the characteristic context.

Now we are in a position to formalize the framework of the decision procedure for Reachability(CCS$^!$, $\doteq$) where $\doteq$ can be either $\equiv$ or $\dot{\equiv}$.

Given two processes $P, Q \in \mathcal{P}_{\mathrm{CCS}^!}$. We want to decide whether $Q \backsimeq \in \mathtt{Drv}(P)$. At first, P can be converted to $\widehat{P}$ which is standard by adding different tags to every subprocess of P just under replicator. After that, we need to convert Q to $\widehat{Q}$ and confirm that $Q \backsimeq \in \mathtt{Drv}(P)$ if and only if $\widehat{Q} \backsimeq \in \mathtt{Drv}(\widehat{P})$. This is done by rewriting Q to some Q' via the structural congruence laws and then adding tags on Q' by guessing! By using some rewriting strategy (eliminating redundant $\mathbf{0}$ when possible), we insure that this can be done algorithmically.

For example, consider process P and Q in Section 3.1. P can be converted into

$$\widehat{P} \equiv !\langle c.!\langle (a.!\langle P_{\mathtt{u}_1}\rangle_{\mathtt{u}_1} + b.!\langle P_{\mathtt{u}_2}\rangle_{\mathtt{u}_2}) \parallel !\langle P_{\mathtt{u}_3}\rangle_{\mathtt{u}_3}\rangle_{\mathtt{v}_2}\rangle_{\mathtt{v}_1}.$$

After that, Q can be converted to $\widehat{Q} \equiv \mathbf{C}'[P_{\mathtt{v}_1}, P_{\mathtt{v}_2}, P_{\mathtt{u}_1}, P_{\mathtt{u}_2}, P_{\mathtt{u}_3}]$, by guessing the position of tags, with

$$\mathbf{C}'[\underline{\mathtt{v}_1}, \underline{\mathtt{v}_2}, \underline{\mathtt{u}_1}, \underline{\mathtt{u}_2}, \underline{\mathtt{u}_3}] = \underline{\mathtt{v}_1} \parallel \langle \underline{\mathtt{v}_2} \parallel \langle \underline{\mathtt{u}_1} \parallel \underline{\mathtt{u}_3}\rangle_{\mathtt{v}_2}\rangle_{\mathtt{v}_1} \parallel \langle \underline{\mathtt{v}_2} \parallel \langle \underline{\mathtt{u}_2} \parallel \langle P'_{\mathtt{u}_2}\rangle_{\mathtt{u}_2} \parallel \underline{\mathtt{u}_3}\rangle_{\mathtt{v}_2}\rangle_{\mathtt{v}_1}$$

where $P_{\mathtt{u}_2} \xrightarrow{\lambda} P'_{\mathtt{u}_2}$. Here we use the representation by characteristic context. Note also the number of active occurrences of $\mathtt{u}_1, \mathtt{u}_2, \mathtt{u}_3$ in $\mathbf{C}'$ are *one*, *one*, *two*, respectively.

The key assertion here is that through the transitions from $\widehat{P}$ to $\widehat{Q}$, $\widehat{P}$ can never transit to the following $\mathbf{C}''[P_{\mathtt{v}_1}, P_{\mathtt{v}_2}, P_{\mathtt{u}_1}, P_{\mathtt{u}_3}]$ with

$$\mathbf{C}''[\underline{\mathtt{v}_1}, \underline{\mathtt{v}_2}, \underline{\mathtt{u}_1}, \underline{\mathtt{u}_3}] = \underline{\mathtt{v}_1} \parallel \langle \underline{\mathtt{v}_2} \parallel \langle \underline{\mathtt{u}_1} \parallel \underline{\mathtt{u}_3}\rangle_{\mathtt{v}_2} \parallel \langle \underline{\mathtt{u}_1} \parallel \underline{\mathtt{u}_3}\rangle_{\mathtt{v}_2}\rangle_{\mathtt{v}_1}$$

in which active occurrences of $\mathtt{u}_1$ is more than once.

In the following, we always assume that P is standard. The intuition described in Section 3.1 is summarized as the next two lemmas.

Lemma 1 (Monotonicity Lemma). *Let P be standard. If $P' \in \mathtt{Drv}(P)$ and $P'' \in \mathtt{Drv}(P')$, then $\mathbf{num}(!\langle P_{\mathtt{v}}\rangle_{\mathtt{v}}, P') \leq \mathbf{num}(!\langle P_{\mathtt{v}}\rangle_{\mathtt{v}}, P'')$.*

Lemma 2 (Bounding Lemma). *Let P be standard, and $Q \in \mathtt{Drv}(P)$. Suppose that $\mathbf{num}(!\langle P_{\mathtt{v}}\rangle_{\mathtt{v}}, Q) = k$, then for every processes R which is an intermediate process during the evolution from P to Q, $\mathbf{num}(!\langle P_{\mathtt{v}}\rangle_{\mathtt{v}}, R) \leq k$.*

Based on Lemma 1 and Lemma 2, a Labelled Petri Net can be constructed. Within this net, we can solve the reachability problem for $\mathrm{CCS}^!$.

Theorem 1. Reachability($\mathrm{CCS}^!$, $\equiv$) *and* Reachability($\mathrm{CCS}^!$, $\dot{\equiv}$) *are both decidable.*

Before ending this section, we define the *component* of a context, which will be used in the formal construction of the Labelled Petri Net in Section 4.

Definition 6. *Let P be in standard form and $\mathbf{C}$ be the characteristic context of P. The* component *of $\mathbf{C}$, denoted by $\mathtt{Comp}(\mathbf{C})$, is defined inductively:*

$$\begin{aligned}
\mathtt{Comp}(\underline{\mathtt{v}}) &\stackrel{\text{def}}{=} \{\underline{\mathtt{v}}\} \\
\mathtt{Comp}(\sum_{i=1}^{n} \lambda_i.\mathbf{C}_i) &\stackrel{\text{def}}{=} \{\sum_{i=1}^{n} \lambda_i.\mathbf{C}_i\} \cup \bigcup_{i=1}^{n} \mathtt{Comp}(\mathbf{C}_i) \\
\mathtt{Comp}(\mathbf{C}_1 \parallel \mathbf{C}_2) &\stackrel{\text{def}}{=} \{\mathbf{C}_1' \parallel \mathbf{C}_2' \mid \mathbf{C}_1' \in \mathtt{Comp}(\mathbf{C}_1), \mathbf{C}_2' \in \mathtt{Comp}(\mathbf{C}_2)\} \\
\mathtt{Comp}((a)\mathbf{C}) &\stackrel{\text{def}}{=} \{(a)\mathbf{C}' \mid \mathbf{C}' \in \mathtt{Comp}(\mathbf{C})\} \\
\mathtt{Comp}(\langle \mathbf{C} \rangle_{\mathtt{v}}) &\stackrel{\text{def}}{=} \{\langle \mathbf{C}' \rangle_{\mathtt{v}} \mid \mathbf{C}' \in \mathtt{Comp}(\mathbf{C})\}
\end{aligned}$$

Note that $\mathtt{Comp}(\mathbf{C})$ is very similar to that of $\mathtt{Sub}(P)$ in [6,7]. In $\mathtt{Sub}(P)$, the localization operators are completely neglected. When localization is taken into account, the defining equation in the case $\mathbf{C}_1 \parallel \mathbf{C}_2$ also need to be modified. Since we will treat the replication operator inductively, there is no need to define $\mathtt{Comp}(!\,P)$. Intuitively, $\mathtt{Comp}(\mathbf{C})$ is understood as the finite set of possible subprocesses in $\mathtt{Drv}(P)$ which is not produced from replication.

4 The Construction of Petri Nets

This section is devoted to the technical part of the construction of Labelled Petri Net $N = (S, \mathcal{A}_\diamond, \rightsquigarrow, \mathbf{m}_{\text{init}})$ from both the source process P and the target process Q. $\mathcal{A}_\diamond$ is the set $\mathcal{A} \cup \{\diamond\}$ in which $\diamond$ is an auxiliary label which appears in N and does not act as a transition label of P. The construction is inductive. For every subprocess of form $!\,\langle P_{\mathtt{v}} \rangle_{\mathtt{v}}$, a Labelled Petri Net $N_{\mathtt{v}} = (S_{\mathtt{v}}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathtt{v}}, \mathbf{m}_{\mathtt{v},\text{init}})$ is constructed. Usually, a place of $N_{\mathtt{v}}$ corresponds to a context of $P_{\mathtt{v}}$. Thus a partial function $\mathtt{Ctxt} : S_{\mathtt{v}} \rightharpoonup \mathcal{C}$ is maintained during the construction. By means of $\mathtt{Ctxt}$ and the inductive procedure, we can compute $\mathtt{Proc}(N_{\mathtt{v}})$, a subprocess of $P_{\mathtt{v}}$, from a given place or marking of $N_{\mathtt{v}}$. The function $\mathtt{Ctxt}$ will be used to translate Q to a marking of N in Section 5. The formal definition of $\mathtt{Proc}$ is omitted since it can be obtained from $\mathtt{Ctxt}$ and the inductive construction.

The construction of N includes three steps: *base step*, *induction step*, and *final step*.

4.1 Base Step

In the base step, we treat the process in the form $!\,\langle P_{\mathtt{v}} \rangle_{\mathtt{v}}$ with $P_{\mathtt{v}}$ replication free. In this special case, $P_{\mathtt{v}}$ is the same as $\mathbf{C}_{\mathtt{v}}$, a context containing no tags. The net for $!\,\langle \mathbf{C}_{\mathtt{v}} \rangle_{\mathtt{v}}$ is $N_{\mathtt{v}} = (S_{\mathtt{v}}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathtt{v}}, \mathbf{m}_{\mathtt{v},\text{init}})$ defined as follows. The place set

$$S_{\mathtt{v}} = [!\,\mathbf{C}_{\mathtt{v}}]_{\mathtt{v}} \cup \{\, [\mathbf{C}]_{\mathtt{v}} \mid \mathbf{C} \in \mathtt{Comp}(\mathbf{C}_{\mathtt{v}})\}.$$

and $\mathtt{Ctxt}([!\,\mathbf{C}_{\mathtt{v}}]_{\mathtt{v}}) = !\,\langle \mathbf{C}_{\mathtt{v}} \rangle_{\mathtt{v}}$, and $\mathtt{Ctxt}([\mathbf{C}]_{\mathtt{v}}) = \langle \mathbf{C} \rangle_{\mathtt{v}}$ if $\mathbf{C} \in \mathtt{Comp}(\mathbf{C}_{\mathtt{v}})$. The initial marking

$$\mathbf{m}_{\mathtt{v},\text{init}} = [!\,\mathbf{C}_{\mathtt{v}}]_{\mathtt{v}}.$$

1. If $\mathbf{C}_{\mathsf{v}} \xrightarrow{\lambda} \mathbf{C}'$, we have rule

$$[!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}} [!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}\ [\mathbf{C}']_{\mathsf{v}} \tag{B1}$$

2. If $\mathbf{C} \xrightarrow{\lambda} \mathbf{C}'$, we have rule

$$[\mathbf{C}]_{\mathsf{v}} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}} [\mathbf{C}']_{\mathsf{v}} \tag{B2}$$

3. If $\mathbf{m}_1 \overset{l}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}_1'$ and $\mathbf{m}_2 \overset{\bar{l}}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}_2'$ are rules defined by (B1) and (B2), we have rule

$$\mathbf{m}_1\ \mathbf{m}_2 \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}_1'\ \mathbf{m}_2' \tag{B3}$$

Fig. 3. Rules for the Base Step

The labelled transition rules $\rightsquigarrow_{\mathsf{v}}$ is defined by the rules in Fig. 3.

Rule (B1) deals with the case that $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}} \xrightarrow{\lambda} !\langle P_{\mathsf{v}}\rangle_{\mathsf{v}} \parallel \langle P_{\mathsf{v}}'\rangle_{\mathsf{v}}$ caused by $\langle P_{\mathsf{v}}\rangle_{\mathsf{v}} \xrightarrow{\lambda} \langle P'\rangle_{\mathsf{v}}$. The derivatives of $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}}$ must be of the form $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}} \parallel \langle P_1\rangle_{\mathsf{v}} \parallel \ldots \parallel \langle P_m\rangle_{\mathsf{v}}$, in which every $P_r (1 \leq r \leq m)$ can be represented as some $\mathbf{C} \in \mathtt{Comp}(\mathbf{C}_{\mathsf{v}})$. Rule (B2) deals with the behaviours of $\langle P_r\rangle_{\mathsf{v}}$'s. Rule (B3) deals with the interaction between two $\langle P_r\rangle_{\mathsf{v}}$'s, or between $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}}$ and $\langle P_r\rangle_{\mathsf{v}}$.

4.2 Induction Step

In the induction step, we treat the process in the form $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}}$. The characteristic context of P_{v} is $\mathbf{C}_{\mathsf{v}}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$, and $P_{\mathsf{v}} = \mathbf{C}_{\mathsf{v}}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}, \ldots, P_{\mathsf{v}_n}]$. Notice also that the base step is the special case of the induction step.

By hypothesis, the Labelled Petri Nets $N_{\mathsf{v}_i} = (S_{\mathsf{v}_i}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_i}, \mathbf{m}_{\mathsf{v}_i,\mathrm{init}})$ has already been constructed for every $!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}$. From these nets, we shall define $N_{\mathsf{v}} = (S_{\mathsf{v}}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}}, \mathbf{m}_{\mathsf{v},\mathrm{init}})$, the net for $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}}$.

Let k_i be $\mathbf{num}(!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}, Q)$. According to the Bounding Lemma, if $Q \in \mathtt{Drv}(P)$, $\mathbf{num}(!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}, R) \leq k_i$ for any intermediate processes R. Because of that, we need k_i disjoint copies of N_{v_i}, named $N_{\mathsf{v}_i,j_i} = (S_{\mathsf{v}_i,j_i}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_i,j_i}, \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i})$ for $j_i = 1, 2, \ldots, k_i$. During the evolution of $!\langle P_{\mathsf{v}}\rangle_{\mathsf{v}}$, whenever a certain $!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}$ come to be active, one of the copies of N_{v_i} is triggered, and the corresponding index j_i is consumed. In this way, N_{v} is constructed, which can partially mimic the process $!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i}$.

Let $\mathbf{C}_{\mathsf{v}}'[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ be a derivative of $\mathbf{C}_{\mathsf{v}}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ in which a certain $\underline{\mathsf{v}_i}$ becomes active. Then one of the k_i copies of N_{v_i} is designated to this active tag. In this case we need to record which copy is designated to a given tag. Thus we need a function $\mathbf{lk}$ which maps every v_i to a number $\mathbf{lk}(\mathsf{v}_i) \in \{\perp\} \cup \{1, 2, \ldots, k_i\}$. If $\mathbf{lk}(\mathsf{v}_i) = j_i$, then the copy with index j_i is designated to tag $\underline{\mathsf{v}_i}$ in the context. If $\mathbf{lk}(\mathsf{v}_i) = \perp$, it means no copy of N_{v_i} has been designated to the tag $\underline{\mathsf{v}_i}$. We will use $\mathbf{lk}_\perp$ to indicate the function with $\mathbf{lk}_\perp(\mathsf{v}_i) = \perp$ for every v_i.

Now we begin to describe the definition of $N_{\mathsf{v}} = (S_{\mathsf{v}}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}}, \mathbf{m}_{\mathsf{v},\mathrm{init}})$. The place set

$$S_{\mathsf{v}} = [\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \cup \bigcup_{\mathbf{lk}} \{\,[\mathbf{C}]^{\mathbf{lk}}_{\mathsf{v}} \mid \mathbf{C} \in \mathtt{Comp}(\mathbf{C}_{\mathsf{v}})\} \cup \bigcup_{i=1}^{n} \bigcup_{j_i=1}^{k_i} \{\mathsf{R}^{j_i}_{\mathsf{v}_i}\} \cup \bigcup_{i=1}^{n} \bigcup_{j_i=1}^{k_i} \{S_{\mathsf{v}_i,j_i}\}.$$

and $\mathtt{Ctxt}([\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}) = !\,\langle \mathbf{C}_{\mathsf{v}} \rangle_{\mathsf{v}}$, $\mathtt{Ctxt}([\mathbf{C}]^{\mathbf{lk}}_{\mathsf{v}}) = \langle \mathbf{C} \rangle_{\mathsf{v}}$ if $\mathbf{C} \in \mathtt{Comp}(\mathbf{C}_{\mathsf{v}})$, $\mathtt{Ctxt}(\mathsf{R}^{j_i}_{\mathsf{v}_i}) = \mathtt{Ctxt}(S_{\mathsf{v}_i,j_i}) = \mathbf{0}$. The places $\mathsf{R}^{j_i}_{\mathsf{v}_i}$'s act as the resources. Whenever a certain copy of N_{v_i}, say N_{v_i,j_i}, is triggered, the corresponding $\mathsf{R}^{j_i}_{\mathsf{v}_i}$'s are consumed. Meanwhile, the superscript $\mathbf{lk}$ is changed to $\mathbf{lk}[\mathsf{v}_i \mapsto j_i]_{i \in I}$ whose value at v_i, originally $\perp$, is changed to j_i. The initial marking

$$\mathbf{m}_{\mathsf{v},\mathrm{init}} = [\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \prod_{i=1}^{n} \prod_{j_i=1}^{k_i} \mathsf{R}^{j_i}_{v_i}.$$

The labelled transition rules $\rightsquigarrow_{\mathsf{v}}$ is defined by the rules in Fig. 4.

Now we explain the rules in Fig. 4.

The initial process $!\,\langle P_{\mathsf{v}} \rangle_{\mathsf{v}}$ is interpreted as the special marking $[\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}$. The behaviour of $!\,\langle P_{\mathsf{v}} \rangle_{\mathsf{v}}$ is specified by the semantic rules Replication. That is, the transition $!\,\langle P_{\mathsf{v}} \rangle_{\mathsf{v}} \xrightarrow{\lambda} !\,\langle P_{\mathsf{v}} \rangle_{\mathsf{v}} \parallel \langle P'_{\mathsf{v}} \rangle_{\mathsf{v}}$ caused by $\langle P_{\mathsf{v}} \rangle_{\mathsf{v}} \xrightarrow{\lambda} \langle P' \rangle_{\mathsf{v}}$. Notice that P_{v} is $\mathbf{C}_{\mathsf{v}}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}, \ldots, P_{\mathsf{v}_n}]$, in which every subprocess $!\,\langle P_{\mathsf{v}_i} \rangle_{\mathsf{v}_i}$ has been interpreted as the initial marking of an arbitrary copy of N_{v_i}. Now the transitions of P_{v} have four possibilities — 1a, 1b, 1c, and 1d. In the case 1a, the transition of $P_{\mathsf{v}} \xrightarrow{\lambda} P'$ is caused by $\mathbf{C}_{\mathsf{v}} \xrightarrow{\lambda} \mathbf{C}'$, and P' is $\mathbf{C}'[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}, \ldots, P_{\mathsf{v}_n}]$. After this transition, some subprocesses $!\,\langle P_{\mathsf{v}_i} \rangle_{\mathsf{v}_i}$ may be active in P', and for every active subprocess, one of N_{v_i,j_i} is attached to the interpretation of P', and the corresponding $\mathsf{R}^{j_i}_{\mathsf{v}_i}$'s are consumed. Now the process $\langle P'_{\mathsf{v}} \rangle_{\mathsf{v}}$ is interpreted as $[\mathbf{C}']^{\mathbf{lk}_\perp[\mathsf{v}_i \mapsto j_i]_{i \in I}}_{\mathsf{v}} \prod_{i \in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i}$. Thus we have rule (I1a). Pay attention that the attached 'subnets' can not evolve by the labelled transition rules of themselves. However, these rules are used to produce labelled transition rules of N_{v}. In the case 1b, $P_{\mathsf{v}} \xrightarrow{\lambda} P'$ is caused by one of the active subprocess $!\,\langle P_{\mathsf{v}_h} \rangle_{\mathsf{v}_h} \xrightarrow{\lambda} R$. In this case, we have $\mathbf{C}_{\mathsf{v}}\{\,!\,\langle P_{\mathsf{v}_h} \rangle_{\mathsf{v}_h}/\mathsf{v}_h\} \xrightarrow{\lambda} \mathbf{C}_{\mathsf{v}}\{R/\mathsf{v}_h\}$ ($\mathbf{C}_{\mathsf{v}}$ is unchanged for only guarded choices are concerned). By induction, $!\,\langle P_{\mathsf{v}_h} \rangle_{\mathsf{v}_h} \xrightarrow{\lambda} R$ is interpreted by a transition rule $\mathbf{m}_{\mathsf{v}_h,\mathrm{init}} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$ of N_{v_h}. By the same argument of 1a, we have rule (I1b). The case 1c treats the situation that $P_{\mathsf{v}} \xrightarrow{\tau} P'$ is caused by interaction between two active subprocess $!\,\langle P_{\mathsf{v}_h} \rangle_{\mathsf{v}_h}$ and $!\,\langle P_{\mathsf{v}_g} \rangle_{\mathsf{v}_g}$. The case 1d treats the situation that $P_{\mathsf{v}} \xrightarrow{\tau} P'$ is caused by interaction between one subprocess $!\,\langle P_{\mathsf{v}_h} \rangle_{\mathsf{v}_h}$ and the environment $\mathbf{C}_{\mathsf{v}}$.

The derivatives of $!\,\langle P_{\mathsf{v}} \rangle_{\mathsf{v}}$ must be of the form $!\,\langle P_{\mathsf{v}} \rangle_{\mathsf{v}} \parallel \langle P_1 \rangle_{\mathsf{v}} \parallel \ldots \parallel \langle P_m \rangle_{\mathsf{v}}$, in which every $P_r (1 \leq r \leq m)$ can be represented as $\mathbf{C}\{R_i/\mathsf{v}_i\}_{i=1}^{n}$ where $R_i \in \mathtt{Drv}(!\,\langle P_{\mathsf{v}_i} \rangle_{\mathsf{v}_i})$. The cases 2a – 2d in Fig. 4 deal with the behaviours of $\langle P_r \rangle_{\mathsf{v}}$. The process $\langle P_r \rangle_{\mathsf{v}}$ is interpreted as one of the $[\mathbf{C}]^{\mathbf{lk}}_{\mathsf{v}}$ attached by certain markings of N_{v_i,j_i} for every i satisfying $\mathbf{C} \rhd \mathsf{v}_i$. There are also four possibilities for transitions

1a. If $\mathbf{C}_{\mathsf{v}} \xrightarrow{\lambda} \mathbf{C}'$, $\mathtt{Act}(\mathbf{C}') = \{\mathsf{v}_i\}_{i\in I}$, we have rule

$$[\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \prod_{i\in I} \mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}} [\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}\ [\mathbf{C}']_{\mathsf{v}}^{\mathbf{lk}_\perp[\mathsf{v}_i\mapsto j_i]_{i\in I}} \prod_{i\in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i} \tag{I1a}$$

for every j_i such that $1 \le j_i \le k_i$.

1b. If $\mathbf{C}_{\mathsf{v}}\{\lambda.\mathbf{0}/\mathsf{v}_h\} \xrightarrow{\lambda} \mathbf{C}_{\mathsf{v}}\{\mathbf{0}/\mathsf{v}_h\}$, $\mathtt{Act}(\mathbf{C}_{\mathsf{v}}) = \{\mathsf{v}_i\}_{i\in I}$, and $\mathbf{m}_{\mathsf{v}_h,\mathrm{init}} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, we have rule

$$[\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \prod_{i\in I} \mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}} [\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}\ [\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}^{\mathbf{lk}_\perp[\mathsf{v}_i\mapsto j_i]_{i\in I}}\ \mathbf{m}'_{\mathsf{v}_h,j_h} \prod_{\substack{i\in I\\ i\neq h}} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i} \tag{I1b}$$

for every j_i such that $1 \le j_i \le k_i$.

1c. If $\mathbf{C}_{\mathsf{v}}\{l.\mathbf{0}/\mathsf{v}_h, \bar{l}.\mathbf{0}/\mathsf{v}_g\} \xrightarrow{\tau} \mathbf{C}_{\mathsf{v}}\{\mathbf{0}/\mathsf{v}_h, \mathbf{0}/\mathsf{v}_g\}$, $\mathtt{Act}(\mathbf{C}_{\mathsf{v}}) = \{\mathsf{v}_i\}_{i\in I}$, $\mathbf{m}_{\mathsf{v}_h,\mathrm{init}} \overset{l}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, and $\mathbf{m}_{\mathsf{v}_g,\mathrm{init}} \overset{\bar{l}}{\rightsquigarrow}_{\mathsf{v}_g} \mathbf{m}'_{\mathsf{v}_g}$, we have rule

$$[\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \prod_{i\in I} \mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} [\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}\ [\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}^{\mathbf{lk}_\perp[\mathsf{v}_i\mapsto j_i]_{i\in I}}\ \mathbf{m}'_{\mathsf{v}_h,j_h}\ \mathbf{m}'_{\mathsf{v}_g,j_g} \prod_{\substack{i\in I\\ i\neq h,g}} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i} \tag{I1c}$$

for every j_i such that $1 \le j_i \le k_i$.

1d. If $\mathbf{C}_{\mathsf{v}}\{l.\mathbf{0}/\mathsf{v}_h\} \xrightarrow{\tau} \mathbf{C}'\{\mathbf{0}/\mathsf{v}_h\}$, $\mathtt{Act}(\mathbf{C}') = \{\mathsf{v}_i\}_{i\in I}$, and $\mathbf{m}_{\mathsf{v}_h,\mathrm{init}} \overset{l}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, we have rule

$$[\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}} \prod_{i\in I} \mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} [\,!\,\mathbf{C}_{\mathsf{v}}]_{\mathsf{v}}\ [\mathbf{C}']_{\mathsf{v}}^{\mathbf{lk}_\perp[\mathsf{v}_i\mapsto j_i]_{i\in I}}\ \mathbf{m}'_{\mathsf{v}_h,j_h} \prod_{\substack{i\in I\\ i\neq h}} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i} \tag{I1d}$$

for every j_i such that $1 \le j_i \le k_i$.

2a. If $\mathbf{C} \xrightarrow{\lambda} \mathbf{C}'$, $\mathtt{Act}(\mathbf{C}') - \mathtt{Act}(\mathbf{C}) = \{\mathsf{v}_i\}_{i\in I}$, we have rule

$$[\mathbf{C}]_{\mathsf{v}}^{\mathbf{lk}} \prod_{i\in I} \mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}} [\mathbf{C}']_{\mathsf{v}}^{\mathbf{lk}[\mathsf{v}_i\mapsto j_i]_{i\in I}} \prod_{i\in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i} \tag{I2a}$$

for every j_i such that $1 \le j_i \le k_i$, and for every $\mathbf{lk}$ such that $\mathbf{lk}(\mathsf{v}_i) = \perp$ for $i \in I$.

2b. If $\mathbf{C}\{\lambda.\mathbf{0}/\mathsf{v}_h\} \xrightarrow{\lambda} \mathbf{C}\{\mathbf{0}/\mathsf{v}_h\}$, and $\mathbf{m}_{\mathsf{v}_h} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, we have rule

$$[\mathbf{C}]_{\mathsf{v}}^{\mathbf{lk}}\ \mathbf{m}_{\mathsf{v}_h,j_h} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}} [\mathbf{C}]_{\mathsf{v}}^{\mathbf{lk}}\ \mathbf{m}'_{\mathsf{v}_h,j_h} \tag{I2b}$$

for every $\mathbf{lk}$ such that $\mathbf{lk}(\mathsf{v}_h) = j_h$.

2c. If $\mathbf{C}\{l.\mathbf{0}/\mathsf{v}_h, \bar{l}.\mathbf{0}/\mathsf{v}_g\} \xrightarrow{\tau} \mathbf{C}\{\mathbf{0}/\mathsf{v}_h, \mathbf{0}/\mathsf{v}_g\}$, $\mathbf{m}_{\mathsf{v}_h} \overset{l}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, and $\mathbf{m}_{\mathsf{v}_g} \overset{\bar{l}}{\rightsquigarrow}_{\mathsf{v}_g} \mathbf{m}'_{\mathsf{v}_g}$, we have rule

$$[\mathbf{C}]_{\mathsf{v}}^{\mathbf{lk}}\ \mathbf{m}_{\mathsf{v}_h,j_h}\ \mathbf{m}_{\mathsf{v}_g,j_g} \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} [\mathbf{C}]_{\mathsf{v}}^{\mathbf{lk}}\ \mathbf{m}'_{\mathsf{v}_h,j_h}\ \mathbf{m}'_{\mathsf{v}_g,j_g} \tag{I2c}$$

for every $\mathbf{lk}$ such that $\mathbf{lk}(\mathsf{v}_h) = j_h$ and $\mathbf{lk}(\mathsf{v}_h) = j_g$.

2d. If $\mathbf{C}\{l.\mathbf{0}/\mathsf{v}_h\} \xrightarrow{\tau} \mathbf{C}'\{\mathbf{0}/\mathsf{v}_h\}$, $\mathtt{Act}(\mathbf{C}') - \mathtt{Act}(\mathbf{C}) = \{\mathsf{v}_i\}_{i\in I}$, and $\mathbf{m}_{\mathsf{v}_h} \overset{l}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, we have rule

$$[\mathbf{C}]_{\mathsf{v}}^{\mathbf{lk}}\ \mathbf{m}_{\mathsf{v}_h,j_h} \prod_{i\in I} \mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} [\mathbf{C}']_{\mathsf{v}}^{\mathbf{lk}[\mathsf{v}_i\mapsto j_i]_{i\in I}}\ \mathbf{m}'_{\mathsf{v}_h,j_h} \prod_{i\in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init},j_i} \tag{I2d}$$

for every j_i such that $1 \le j_i \le k_i$,, and for every $\mathbf{lk}$ such that $\mathbf{lk}(\mathsf{v}_h) = j_h$ and $\mathbf{lk}(\mathsf{v}_i) = \perp$ for $i \in I$.

3. If $\mathbf{m}_1 \overset{l}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}_1'$ and $\mathbf{m}_2 \overset{\bar{l}}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}_2'$ are rules defined by (I1a)–(I1d) and (I2a)–(I2d), we have rule

$$\mathbf{m}_1\ \mathbf{m}_2 \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}_1'\ \mathbf{m}_2' \tag{I3}$$

4. We have rules

$$\mathsf{R}_{\mathsf{v}_i}^{j_i} \overset{\diamond}{\rightsquigarrow}_{\mathsf{v}} \epsilon \tag{I4}$$

for every v_i and for every j_i satisfying $1 \leq j_i \leq k_i$.

Fig. 4. Rules for the Induction Step

of $\langle P_r \rangle_{\mathsf{v}}$. In the case 2a, $\langle P_r \rangle_{\mathsf{v}} \xrightarrow{\lambda} P'$ is caused by $\mathbf{C} \xrightarrow{\lambda} \mathbf{C}'$. If some new tags, say $\{\mathsf{v}_i\}_{i \in I}$, become active, the copies of N_{v_i}'s are attached in the same way. This leads to rule (I2a). Case 2b deals with the situation that $\langle P_r \rangle_{\mathsf{v}} \xrightarrow{\lambda} P'$ is caused by $R_h \xrightarrow{\lambda} R_h'$. In the case 2c, $\langle P_r \rangle_{\mathsf{v}} \xrightarrow{\tau} P'$ is caused by interaction between R_h and R_g, while in the case 2d, the transition $\langle P_r \rangle_{\mathsf{v}} \xrightarrow{\tau} P'$ is caused by interaction between R_h and $\mathbf{C}$.

Rule (I3) deals with the case that interaction happens between $\langle P_r \rangle_{\mathsf{v}}$'s, or between $!\langle P_{\mathsf{v}} \rangle_{\mathsf{v}}$ and $\langle P_r \rangle_{\mathsf{v}}$.

Rule (I4) says that the resource processes $\mathsf{R}_{v_i}^{j_i}$'s can be consumed without side-effect at any moment. The special label $\diamond$ is used here.

The correctness of the construction in base step and induction step is stated in the next two lemmas.

Lemma 3. *If a marking* $\mathbf{m}$ *of* N_{v} *is reachable from* $\mathbf{m}_{\mathsf{v},\mathrm{init}}$*, then* $\mathtt{Proc}(\mathbf{m})$ *is reachable from* $!\langle \mathbf{P}_{\mathsf{v}} \rangle_{\mathsf{v}}$.

Lemma 4. *If* $P' \equiv !\langle P_{\mathsf{v}} \rangle_{\mathsf{v}} \parallel \langle P_1 \rangle_{\mathsf{v}} \parallel \langle P_2 \rangle_{\mathsf{v}} \parallel \ldots \parallel \langle P_n \rangle_{\mathsf{v}}$ *is reachable from* $!\langle P_{\mathsf{v}} \rangle_{\mathsf{v}}$*, and* $\mathbf{num}(!\langle P_{\mathsf{v}_i} \rangle_{\mathsf{v}_i}, P') \leq \mathbf{num}(!\langle P_{\mathsf{v}_i} \rangle_{\mathsf{v}_i}, Q)$*, then there is a marking* $\mathbf{m}$ *of* N_{v} *such that* $\mathtt{Proc}(\mathbf{m}) \equiv P'$ *and* $\mathbf{m}$ *is reachable from* $\mathbf{m}_{\mathsf{v},\mathrm{init}}$.

4.3 Final Step

In the final step, we treat the process P in the form $\mathbf{C}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}, \ldots, P_{\mathsf{v}_n}]$, in which $\mathbf{C}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}, \ldots, \underline{\mathsf{v}_n}]$ is the characteristic context of P. The final step is a simplified version of the induction step for the absence of outermost replication operator.

By the induction step, the Labelled Petri Net $N_{\mathsf{v}_i} = (S_{\mathsf{v}_i}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_i}, \mathbf{m}_{\mathsf{v}_i,\mathrm{init}})$ has already been constructed for every $!\langle P_{\mathsf{v}_i} \rangle_{\mathsf{v}_i}$. In this final step, only *one* copy of N_{v_i} is needed for every v_i appearing in the characteristic context of P. In the required Labelled Petri Net $N = (S, \mathcal{A}_\diamond, \rightsquigarrow, \mathbf{m}_{\mathrm{init}})$, the place set

$$S = \{\, [\mathbf{C}'] \mid \mathbf{C}' \in \mathtt{Comp}(\mathbf{C}) \} \cup \bigcup_{i=1}^{n} \{S_{\mathsf{v}_i}\}.$$

1a. If $\mathbf{C} \xrightarrow{\lambda} \mathbf{C}'$, $\mathtt{Act}(\mathbf{C}') - \mathtt{Act}(\mathbf{C}) = \{\mathsf{v}_i\}_{i\in I}$, we have rule

$$[\mathbf{C}] \overset{\lambda}{\rightsquigarrow} [\mathbf{C}'] \prod_{i\in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init}} \tag{F1a}$$

1b. If $\mathbf{C}\{\lambda.\mathbf{0}/\mathsf{v}_h\} \xrightarrow{\lambda} \mathbf{C}\{\mathbf{0}/\mathsf{v}_h\}$, and $\mathbf{m}_{\mathsf{v}_h} \overset{\lambda}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, we have rule

$$[\mathbf{C}]\ \mathbf{m}_{\mathsf{v}_h} \overset{\lambda}{\rightsquigarrow} [\mathbf{C}]\ \mathbf{m}'_{\mathsf{v}_h} \tag{F1b}$$

1c. If $\mathbf{C}\{l.\mathbf{0}/\mathsf{v}_h, \bar{l}.\mathbf{0}/\mathsf{v}_g\} \xrightarrow{\tau} \mathbf{C}\{\mathbf{0}/\mathsf{v}_h, \mathbf{0}/\mathsf{v}_g\}$, $\mathbf{m}_{\mathsf{v}_h} \overset{l}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, and $\mathbf{m}_{\mathsf{v}_g} \overset{\bar{l}}{\rightsquigarrow}_{\mathsf{v}_g} \mathbf{m}'_{\mathsf{v}_g}$, we have rule

$$[\mathbf{C}]\ \mathbf{m}_{\mathsf{v}_h}\ \mathbf{m}_{\mathsf{v}_g} \overset{\tau}{\rightsquigarrow} [\mathbf{C}]\ \mathbf{m}'_{\mathsf{v}_h}\ \mathbf{m}'_{\mathsf{v}_g} \tag{F1c}$$

1d. If $\mathbf{C}\{l.\mathbf{0}/\mathsf{v}_h\} \xrightarrow{\tau} \mathbf{C}'\{\mathbf{0}/\mathsf{v}_h\}$, $\mathtt{Act}(\mathbf{C}') - \mathtt{Act}(\mathbf{C}) = \{\mathsf{v}_i\}_{i\in I}$, and $\mathbf{m}_{\mathsf{v}_h} \overset{l}{\rightsquigarrow}_{\mathsf{v}_h} \mathbf{m}'_{\mathsf{v}_h}$, we have rule

$$[\mathbf{C}]\ \mathbf{m}_{\mathsf{v}_h} \overset{\tau}{\rightsquigarrow} [\mathbf{C}']\ \mathbf{m}'_{\mathsf{v}_h} \prod_{i\in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init}} \tag{F1d}$$

2. If $\mathbf{m}_1 \overset{l}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}'_1$ and $\mathbf{m}_2 \overset{\bar{l}}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}'_2$ are rules defined by (F1a)–(F1d), we have rule

$$\mathbf{m}_1\ \mathbf{m}_2 \overset{\tau}{\rightsquigarrow}_{\mathsf{v}} \mathbf{m}'_1\ \mathbf{m}'_2 \tag{F2}$$

Fig. 5. Rules for the Final Step

and $\mathtt{Ctxt}([\mathbf{C}']) = \mathbf{C}$ if $\mathbf{C}' \in \mathtt{Comp}(\mathbf{C})$. The initial marking

$$\mathbf{m}_{\mathrm{init}} = [\mathbf{C}] \prod_{i\in I} \mathbf{m}_{\mathsf{v}_i,\mathrm{init}}.$$

assume that $\mathtt{Act}(\mathbf{C}) = \{\mathsf{v}_i\}_{i\in I}$.

The labelled transition rules of N are summarized in Fig. 5. Since every subnet N_{v_i} is used at most once, we do not need the function **lk** and the auxiliary places such as $\mathsf{R}^{j_i}_{\mathsf{v}_i}$'s. Rule (F1a)–(F1d) is the simplified version of Rule (I2a)–(I2d), and there is no counterpart of Rule (I1a)–(I1d).

5 Deciding Reachability Problem

In Section 4, a Labelled Petri Net $N = (S, \mathcal{A}_\diamond, \rightsquigarrow, \mathbf{m}_{\mathrm{init}})$ is constructed based on both the source process P and the target process Q. P is interpreted as the marking $\mathbf{m}_{\mathrm{init}}$ of N. If $Q \in \mathtt{Drv}(P)$, we need to find in N the marking $\mathbf{m}_Q$ of Q, and confirm that $\mathbf{m}_Q$ is reachable from $\mathbf{m}_{\mathrm{init}}$ if and only if $Q \in \mathtt{Drv}(P)$. This work is accomplished in a top-down fashion by procedure $Trans(Q, N)$ in Fig. 6.

If $\mathbf{C}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}, \ldots, P_{\mathsf{v}_n}]$ and $Q \in \mathtt{Drv}(P)$, then, for some $\mathbf{C}_Q \in \mathtt{Drv}(\mathbf{C})$, Q must be in the form of $\mathbf{C}_Q[\underline{\mathsf{v}_1}, \ldots, \underline{\mathsf{v}_n}]\{O_{\mathsf{v}_i}/\underline{\mathsf{v}_i}\}_{i=1}^n$, where O_{v_i} is of the form

$Trans^!(O_{\mathsf{u}}, N_{\mathsf{u}})$:

1. Find distinct $\mathsf{s}_h \in S_{\mathsf{u}}$ such that $\mathtt{Ctxt}(\mathsf{s}_h) = \mathbf{C}^h_{\mathsf{u}}$, and $\mathsf{s} \in S$ such that $\mathtt{Ctxt}(\mathsf{s}) = \mathbf{C}_u$.
2. **for** each h **for** each $\underline{\mathsf{u}_i}$ active in $\mathbf{C}^h_{\mathsf{u}}$, find arbitrary one copy of N_{u_i}, say $N_{\mathsf{u}_i,j_h} = (S_{\mathsf{u}_i,j_h}, \mathcal{A}_\diamond, \rightsquigarrow_{\mathsf{u}_i,j_h}, \mathbf{m}_{\mathsf{v}_i,\text{init}}, j_h)$ which is constructed for $!\langle P_{\mathsf{u}_i}\rangle_{\mathsf{u}_i}$, and bind s_h with N_{u_i,j_h}. Let $\mathbf{m}^h_{\mathsf{u}_i} = Trans^!(O^j_{\mathsf{u}_i}, N_{\mathsf{u}_i,j_h})$.
3. **return** $\mathsf{s}\ (\prod_h \mathsf{s}_h)\ (\prod_{\mathsf{u}_i,h} \mathbf{m}^h_{\mathsf{u}_i})$.

$Trans(Q, N)$:

1. Find $\mathsf{s} \in S$ in N such that $\mathtt{Ctxt}(\mathsf{s}) = \mathbf{C}_Q$.
2. **for** each $\underline{\mathsf{v}_i}$ active in $\mathbf{C}_Q$, find $N_{\mathsf{v}_i} = (S_{\mathsf{v}_i}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_i}, \mathbf{m}_{\mathsf{v}_i,\text{init}})$. Let $\mathbf{m}_{\mathsf{v}_i} = Trans(O_{\mathsf{v}_i}, N_{\mathsf{v}_i})$
3. **return** $\mathsf{s} \prod_{\mathsf{v}_i} \mathbf{m}_{\mathsf{v}_i}$.

Fig. 6. The target process as a marking

$!\langle P_{\mathsf{v}_i}\rangle_{\mathsf{v}_i} \parallel \prod_h \langle P_{\mathsf{v}_i,h}\rangle_{\mathsf{v}_i}$. When $\mathbf{C}_Q \rhd \underline{\mathsf{v}_i}$, the subprocedure $Trans^!(O_{\mathsf{v}_i}, N_{\mathsf{v}_i})$ is called in order to get the sub-marking of O_{v_i}.

The procedure $Trans^!(O_{\mathsf{u}}, N_{\mathsf{u}})$ aims at the marking for is called, O_{u} must be of the form

$$!\langle \mathbf{C}_{\mathsf{u}}\{P_{\mathsf{u}_i}/\underline{\mathsf{u}_i}\}_{i=1}^m\rangle_{\mathsf{u}} \parallel \prod_h \langle \mathbf{C}^h_{\mathsf{u}}\{O^h_{\mathsf{u}_i}/\underline{\mathsf{u}_i}\}_{i=1}^m\rangle_{\mathsf{u}}$$

After that, $Trans^!$ may be called recursively with parameters getting smaller and smaller depending on the structure of Q. It is worth noting that, in case Q does not have the desired structure, Q cannot be a derivative of P, and in this situation $Trans(Q, N)$ will terminate with no marking returned.

Lemma 5. *If Q is a process for which $Trans(Q, N)$ returns a marking $\mathbf{m}$ successfully, then, $\mathbf{m}$ can be reached from $\mathbf{m}_{\text{init}}$ if and only if $Q \in \mathtt{Drv}(P)$.*

6 An Illustrative Example

In this section, we give an example to illustrate how the Labelled Petri Net N is constructed from a given P and Q, and to illustrate how the algorithm works.

Let

$$\begin{aligned}
P &\stackrel{\text{def}}{=} (b)(\,!\,\langle b.(a)(\,!\,\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel \overline{b})\\
Q &\stackrel{\text{def}}{=} (b)(\,!\,\langle b.(a)(\,!\,\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
&\quad \langle (a)(\,!\,\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
&\quad \langle (a)(\,!\,\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
&\quad \langle (a)(\,!\,\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}})
\end{aligned}$$

At first, we represent P and Q using characteristic contexts. P is represented as $\mathbf{C}_P[P_{\mathsf{u}}]$ in which $\mathbf{C}_P[\underline{\mathsf{u}}] = (b)(\underline{\mathsf{u}} \parallel \overline{b})$ and $P_{\mathsf{u}} \equiv b.(a)(!\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})$. Now P_u is represented as $\mathbf{C}_{\mathsf{u}}[P_{\mathsf{v}_1}, P_{\mathsf{v}_2}]$ in which $\mathbf{C}_{\mathsf{u}}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}]$ is $b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})$ and $P_{\mathsf{v}_1} \equiv a.\overline{b}$, $P_{\mathsf{v}_2} \equiv \overline{a}.\overline{b}$.

The standard representation of Q is $\mathbf{C}_Q[P_{\mathsf{u}}, P_{\mathsf{v}_1}, P_{\mathsf{v}_2}]$ in which

$$\mathbf{C}_Q[\underline{\mathsf{u}}, \underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}] = (b)(\underline{\mathsf{u}} \parallel \langle (a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel \langle (a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel \langle (a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})\rangle_{\mathsf{u}}).$$

The number of active occurrences of $\underline{\mathsf{u}}, \underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}$ in $\mathbf{C}_Q$ is $1, 3, 3$, respectively.

Now we begin to describe the construction of N. For simplicity, only possibly useful places and transitions are stated explicitly here.

In the *base step*, we construct $N_{\mathsf{v}_1} = (S_{\mathsf{v}_1}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_1}, \mathbf{m}_{\mathsf{v}_1,\text{init}})$ and $N_{\mathsf{v}_2} = (S_{\mathsf{v}_2}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_2}, \mathbf{m}_{\mathsf{v}_2,\text{init}})$ for the processes $!\langle P_{\mathsf{v}_1}\rangle_{\mathsf{v}_1} \equiv !\langle a.\overline{b}\rangle_{\mathsf{v}_1}$ and $!\langle P_{\mathsf{v}_2}\rangle_{\mathsf{v}_2} \equiv !\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2}$. The initial marking $\mathbf{m}_{\mathsf{v}_1,\text{init}}$ is $[!\,a.\overline{b}]_{\mathsf{v}_1}$, and $\mathbf{m}_{\mathsf{v}_2,\text{init}}$ is $[!\,\overline{a}.\overline{b}]_{\mathsf{v}_2}$.

Using Rule (B1) and Rule (B2), we have the following transition rules:

$$[!\,a.\overline{b}]_{\mathsf{v}_1} \overset{a}{\rightsquigarrow}_{\mathsf{v}_1} [!\,a.\overline{b}]_{\mathsf{v}_1}\ [\overline{b}]_{\mathsf{v}_1}$$
$$[\overline{b}]_{\mathsf{v}_1} \overset{\overline{b}}{\rightsquigarrow}_{\mathsf{v}_1} \epsilon$$

and

$$[!\,\overline{a}.\overline{b}]_{\mathsf{v}_2} \overset{a}{\rightsquigarrow}_{\mathsf{v}_2} [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2}\ [\overline{b}]_{\mathsf{v}_2}$$
$$[\overline{b}]_{\mathsf{v}_2} \overset{\overline{b}}{\rightsquigarrow}_{\mathsf{v}_2} \epsilon$$

This completes the base step.

In the *induction step*, we construct $N_{\mathsf{u}} = (S_{\mathsf{u}}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{u}}, \mathbf{m}_{\mathsf{u},\text{init}})$ for the process $!\langle P_{\mathsf{u}}\rangle_{\mathsf{u}} \equiv !\langle b.(a)(!\langle a.\overline{b}\rangle_{\mathsf{v}_1} \parallel !\langle \overline{a}.\overline{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}}$. Since both $\underline{\mathsf{v}_1}$ and $\underline{\mathsf{v}_2}$ occur three times in $\mathbf{C}_Q$, three copies of N_{v_1} and N_{v_2} are needed, which are denoted by $N_{\mathsf{v}_i,j_i} = (S_{\mathsf{v}_i,j_i}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{v}_i,j_i}, \mathbf{m}_{\mathsf{v}_i,\text{init},j_i})$ for $i = 1, 2$ and $j_i = 1, 2, 3$.

Recall that $\mathbf{C}_{\mathsf{u}}[\underline{\mathsf{v}_1}, \underline{\mathsf{v}_2}]$ is $b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})$. The initial marking of N_{u} is

$$\mathbf{m}_{\mathsf{u},\text{init}} = [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}} \prod_{j_1=1}^{3} \mathsf{R}_{\mathsf{v}_1}^{j_1} \prod_{j_2=1}^{3} \mathsf{R}_{\mathsf{v}_2}^{j_2}$$

Now we begin to work out the transition rules of N_{u}. A function $\mathbf{lk}$ will be notated as an ordered pair (j_1, j_2), which means that $\mathbf{lk}(\mathsf{v}_1) = j_1$ and $\mathbf{lk}(\mathsf{v}_2) = j_2$. In all the following transition rules, $j_1, j_2, j_1', j_2' \in \{1, 2, 3\}$.

– By Rule (I1a), N_{u} has the following transitions:

$$[!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}_{\mathsf{v}_1}^{j_1}\ \mathsf{R}_{\mathsf{v}_2}^{j_2}$$
$$\overset{b}{\rightsquigarrow}_{\mathsf{u}} [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2}$$

– By Rule (I2c), N_{u} has the following transitions:

$$[(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2}$$
$$\overset{\tau}{\rightsquigarrow}_{\mathsf{u}} [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1}\ [\overline{b}]_{\mathsf{v}_1,j_1}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2}\ [\overline{b}]_{\mathsf{v}_2,j_2}$$

– By Rule (I2b), N_{u} has the following transitions:

$$[(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [\overline{b}]_{\mathsf{v}_1,j_1} \overset{\overline{b}}{\rightsquigarrow}_{\mathsf{u}} [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}$$

and

$$[(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [\overline{b}]_{\mathsf{v}_2,j_2} \overset{\overline{b}}{\rightsquigarrow}_{\mathsf{u}} [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}$$

– By Rule (I3), N_{u} has the following transitions:

$$\begin{aligned}&[!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}_{\mathsf{v}_1}^{j_1'}\ \mathsf{R}_{\mathsf{v}_2}^{j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [\overline{b}]_{\mathsf{v}_1,j_1}\\ \overset{\tau}{\rightsquigarrow}_{\mathsf{u}}\ &[!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1',j_2')}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1'}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\end{aligned}$$

and

$$\begin{aligned}&[!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}_{\mathsf{v}_1}^{j_1'}\ \mathsf{R}_{\mathsf{v}_2}^{j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [\overline{b}]_{\mathsf{v}_2,j_2}\\ \overset{\tau}{\rightsquigarrow}_{\mathsf{u}}\ &[!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1',j_2')}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1'}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\end{aligned}$$

In the *final step*, P is translated to $N = (S, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow, \mathbf{m}_{\text{init}})$, based on the construction of $N_{\mathsf{u}} = (S_{\mathsf{u}}, \mathcal{A} \cup \{\diamond\}, \rightsquigarrow_{\mathsf{u}}, \mathbf{m}_{\mathsf{u},\text{init}})$ in the induction step.

We recall that P is $\mathbf{C}_P[P_{\mathsf{u}}]$ where $\mathbf{C}_P[\underline{\mathsf{u}}] = (b)(\underline{\mathsf{u}} \parallel \overline{b})$. The initial marking of N is

$$\mathbf{m}_{\text{init}} = [(b)(\underline{\mathsf{u}} \parallel \overline{b})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}} \prod_{j_1=1}^{3} \mathsf{R}_{\mathsf{v}_1}^{j_1} \prod_{j_2=1}^{3} \mathsf{R}_{\mathsf{v}_2}^{j_2}$$

The transition rules are described as follows with the same convention.

– By Rule F1d, N has the following transitions:

$$\begin{aligned}&[(b)(\underline{\mathsf{u}} \parallel \overline{b})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}_{\mathsf{v}_1}^{j_1}\ \mathsf{R}_{\mathsf{v}_2}^{j_2}\\ \overset{\tau}{\rightsquigarrow}\ &[(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2}\end{aligned}$$

– By Rule F1b, N has the following transitions:

$$\begin{aligned}&[(b)(\underline{\mathsf{u}})]\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2}\\ \overset{\tau}{\rightsquigarrow}\ &[(b)(\underline{\mathsf{u}})]\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [!\,a.\overline{b}]_{\mathsf{v}_1,j_1}\ [\overline{b}]_{\mathsf{v}_1,j_1}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2}\ [\overline{b}]_{\mathsf{v}_2,j_2}\end{aligned}$$

– By Rule F1b, N also has the following transitions:

$$\begin{aligned}&[(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}_{\mathsf{v}_1}^{j_1'}\ \mathsf{R}_{\mathsf{v}_2}^{j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [\overline{b}]_{\mathsf{v}_1,j_1}\\ \overset{\tau}{\rightsquigarrow}\ &[(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1',j_2')}\\ &[!\,a.\overline{b}]_{\mathsf{v}_1,j_1'}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\end{aligned}$$

and

$$\begin{aligned}&[(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}_{\mathsf{v}_1}^{j_1'}\ \mathsf{R}_{\mathsf{v}_2}^{j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\ [\overline{b}]_{\mathsf{v}_2,j_2}\\ \overset{\tau}{\rightsquigarrow}\ &[(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1',j_2')}\\ &[!\,a.\overline{b}]_{\mathsf{v}_1,j_1'}\ [!\,\overline{a}.\overline{b}]_{\mathsf{v}_2,j_2'}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(j_1,j_2)}\end{aligned}$$

So far, the Labelled Petri Net N is constructed. Next, we can use the algorithm in Fig. 6 to find a translation of Q as a marking of N:

$$\mathbf{m}_Q = [(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}[(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(1,1)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,1}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,1}$$
$$[(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(2,2)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,2}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,2}\ [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(3,3)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,3}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,3}$$

According to the semantics of CCS$^!$, P have the following way to reach Q:

$$\begin{aligned}
& (b)(!\,\langle b.(a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel \bar{b})\\
\xrightarrow{\tau}\ & (b)(!\,\langle b.(a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}})\\
\xrightarrow{\tau}\ & (b)(!\,\langle b.(a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel \langle \bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2} \parallel \langle \bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}})\\
\xrightarrow{\tau}\ & (b)(!\,\langle b.(a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2} \parallel \langle \bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}})\\
\xrightarrow{\tau}\ & (b)(!\,\langle b.(a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}} \parallel\\
& \quad \langle (a)(!\,\langle a.\bar{b}\rangle_{\mathsf{v}_1} \parallel !\,\langle \bar{a}.\bar{b}\rangle_{\mathsf{v}_2})\rangle_{\mathsf{u}})
\end{aligned}$$

This computational path is simulated by N from marking $\mathbf{m}_{\text{init}}$ to $\mathbf{m}_Q$:

$$\begin{aligned}
& [(b)(\underline{\mathsf{u}} \parallel \bar{b})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}^1_{\mathsf{v}_1}\ \mathsf{R}^2_{\mathsf{v}_1}\ \mathsf{R}^3_{\mathsf{v}_1}\ \mathsf{R}^1_{\mathsf{v}_2}\ \mathsf{R}^2_{\mathsf{v}_2}\ \mathsf{R}^3_{\mathsf{v}_2}\\
\overset{\tau}{\leadsto}\ & [(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}^2_{\mathsf{v}_1}\ \mathsf{R}^3_{\mathsf{v}_1}\ \mathsf{R}^2_{\mathsf{v}_2}\ \mathsf{R}^3_{\mathsf{v}_2}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(1,1)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,1}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,1}\\
\overset{\tau}{\leadsto}\ & [(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}^2_{\mathsf{v}_1}\ \mathsf{R}^3_{\mathsf{v}_1}\ \mathsf{R}^2_{\mathsf{v}_2}\ \mathsf{R}^3_{\mathsf{v}_2}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(1,1)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,1}\ [\bar{b}]_{\mathsf{v}_1,1}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,1}\ [\bar{b}]_{\mathsf{v}_2,1}\\
\overset{\tau}{\leadsto}\ & [(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}^3_{\mathsf{v}_1}\ \mathsf{R}^3_{\mathsf{v}_2}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(1,1)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,1}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,1}\ [\bar{b}]_{\mathsf{v}_2,1}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(2,2)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,2}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,2}\\
\overset{\tau}{\leadsto}\ & [(b)(\underline{\mathsf{u}})]\ [!\,b.(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}\ \mathsf{R}^3_{\mathsf{v}_1}\ \mathsf{R}^3_{\mathsf{v}_2}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(1,1)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,1}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,1}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(2,2)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,2}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,2}\\
& \quad [(a)(\underline{\mathsf{v}_1} \parallel \underline{\mathsf{v}_2})]_{\mathsf{u}}^{(3,3)}\ [!\,a.\bar{b}]_{\mathsf{v}_1,3}\ [!\,\bar{a}.\bar{b}]_{\mathsf{v}_2,3}
\end{aligned}$$

7 Concluding Remark

We have presented a deciding procedure of the reachability problem for CCS$^!$. In order to focus on the main argument, the syntax and semantics are simplified:

The rule Replication is REPL1 in [5], while REPL2 is ignored; The guarded choice is used instead of the general choice. The decidability result will not change for such kinds of generalization of CCS$^!$. The reachability problem can also be confined by only considering τ-transitions. This confined problem is also decidable.

The interplay between replication and localization makes CCS$^!$ very expressive. Some basic properties of CCS$^!$ are studied in [5,6,7]. The relative expressiveness of variants of CCS is further studied in [11,10]. It is proved in [6,11] that CCS$^!$ and CCS$^\mu$ are less expressive than CCS$^{\mathrm{Pdef}}$. The two problems left open in [11] are both answered positively in [10], which confirms the existence of an encoding from CCS$^\mu$ to CCS$^!$ that is codivergent branching bisimilar, and the existence of an encoding from CCS$^\mu$ to itself with only guarded recursion. The expressiveness of CCS$^!$ is also studied in [3,4]. A unified approach to the study of relative expressiveness is proposed in [9]. It is shown in [1,2] that CCS$^!$ can express behavioural types for the π-calculus, while several safety properties are still decidable.

Are there more direct ways to decide the reachability problem for CCS$^!$? In literature some formalisms which is more powerful than LPN are studied, for example PRS [20] and wPRS, whose reachability problem is decidable [20,16]. Afterward, for these formalism the reachability of HM Property is also decidable [17]. These facts suggest that encoding CCS$^!$ into PRS or wPRS directly is impossible.

Acknowledgements The author thanks the anonymous referees for their detailed reviews on the previous version of the paper. Their criticisms, questions and suggestions have led to a significant improvement of the paper. He likes to thank professor Yuxi Fu for helpful suggestions, and to thank Hongfei Fu for discussions in the initial stage of this work.

References

1. Lucia Acciai and Michele Boreale. Spatial and behavioral types in the pi-calculus. In *CONCUR*, pages 372–386, 2008.
2. Lucia Acciai and Michele Boreale. Deciding safety properties in infinite-state pi-calculus via behavioural types. In *ICALP (2)*, pages 31–42, 2009.
3. Jesús Aranda, Cinzia Di Giusto, Mogens Nielsen, and Frank D. Valencia. Ccs with replication in the chomsky hierarchy: The expressive power of divergence. In *APLAS*, pages 383–398, 2007.
4. Jesús Aranda, Frank D. Valencia, and Cristian Versari. On the expressive power of restriction and priorities in ccs with replication. In *FOSSACS*, pages 242–256, 2009.
5. Nadia Busi, Maurizio Gabbrielli, and Gianluigi Zavattaro. Replication vs. recursive definitions in channel based calculi. In *ICALP*, pages 133–144, 2003.
6. Nadia Busi, Maurizio Gabbrielli, and Gianluigi Zavattaro. Comparing recursion, replication, and iteration in process calculi. In *ICALP*, pages 307–319, 2004.
7. Nadia Busi, Maurizio Gabbrielli, and Gianluigi Zavattaro. On the expressive power of recursion, replication and iteration in process calculi. *Mathematical Structures in Computer Science*, 19(6):1191–1222, 2009.

8. Nadia Busi and Roberto Gorrieri. Distributed conflicts in communicating systems. In *ECOOP Workshop*, pages 49–65, 1994.
9. Yuxi Fu. Theory of interaction. 2011. *Submitted and downloadable at* `http://basics.sjtu.edu.cn/~yuxi/`,.
10. Yuxi Fu and Hao Lu. On the expressiveness of interaction. *Theor. Comput. Sci.*, 411(11-13):1387–1451, 2010.
11. Pablo Giambiagi, Gerardo Schneider, and Frank D. Valencia. On the expressiveness of infinite behavior and name scoping in process calculi. In *FoSSaCS*, pages 226–240, 2004.
12. Ursula Goltz. CCS and petri nets. In *Semantics of Systems of Concurrent Processes*, pages 334–357, 1990.
13. Chaodong He, Yuxi Fu, and Hongfei Fu. Decidability of behavioral equivalences in process calculi with name scoping. In *FSEN*, 2011.
14. Petr Jancar and Faron Moller. Checking regular properties of petri nets. In *CONCUR*, pages 348–362, 1995.
15. S. Rao Kosaraju. Decidability of reachability in vector addition systems (preliminary version). In *STOC*, pages 267–281, 1982.
16. Mojmír Kretínský, Vojtech Rehák, and Jan Strejcek. Extended process rewrite systems: Expressiveness and reachability. In *CONCUR*, pages 355–370, 2004.
17. Mojmír Kretínský, Vojtech Rehák, and Jan Strejcek. Reachability of hennessy-milner properties for weakly extended prs. In *FSTTCS*, pages 213–224, 2005.
18. Jean-Luc Lambert. A structure to decide reachability in petri nets. *Theor. Comput. Sci.*, 99(1):79–104, 1992.
19. Ernst W. Mayr. An algorithm for the general petri net reachability problem. In *STOC*, pages 238–246, 1981.
20. Richard Mayr. Process rewrite systems. *Inf. Comput.*, 156(1-2):264–286, 2000.
21. Robin Milner. *Communication and Concurrency*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.
22. Robin Milner. *Communicating and Mobile Systems: the π-calculus*. Cambridge University Press, 1999.
23. David Michael Ritchie Park. Concurrency and automata on infinite sequences. In *Theoretical Computer Science*, pages 167–183, 1981.
24. George S. Sacerdote and Richard L. Tenney. The decidability of the reachability problem for vector addition systems (preliminary version). In *STOC*, pages 61–76, 1977.
25. Dirk Taubner. *Finite Representations of CCS and TCSP Programs by Automata and Petri Nets*, volume 369 of *Lecture Notes in Computer Science*. Springer, 1989.