Reversibility in Process Calculi with Nondeterminism and Probabilities

Marco Bernardo and Claudio A. Mezzina

Dipartimento di Scienze Pure e Applicate, Università di Urbino, Urbino, Italy

Abstract. A reversible system features not only forward computations, but also backward computations along which the effects of forward ones can be undone by starting from the last performed action. According to causal reversibility, an executed action can be undone provided that all the actions it caused have been undone already. We investigate causal reversibility in a nondeterministic and probabilistic setting by adapting the framework of Phillips and Ulidowski to define a reversible calculus in which action transitions and probabilistic transitions alternate in the style of Hansson and Jonsson. We show that the calculus meets causal reversibility through a suitable variant of the technique of Lanese, Phillips, and Ulidowski that ensures the proper forward and backward interplay of nondeterminism and probabilities. The use of the calculus is illustrated on a quantum computing example.

1 Introduction

Reversible computing has the potential of achieving lower energy consumption because irreversible manipulation of information must be accompanied by an entropy increase due to heat dissipation [25,2,7,16]. Its applications encompass biochemical reaction modeling [37,38], parallel discrete-event simulation [34,41], fault-tolerant systems [11,48,26,47], concurrent program debugging [17,28], robotics [30], control theory [45], and wireless communications [45].

Reversibility in a computing system has to do with the possibility of reverting actions starting from the last performed one. In a concurrent system there may not be a total order over executed actions, hence the last performed action may not be uniquely identifiable. This led to the introduction of the notion of causal reversibility [10], according to which a previously executed action can be undone provided that all of its consequences, if any, have been undone beforehand.

In the process algebraic setting, two approaches have been developed to deal with causal reversibility. The dynamic one of [10,24] attaches external stack-based memories to process terms so as to store executed actions and discarded subprocesses. A single transition relation is present, where transitions can be labeled with forward or backward actions. In contrast, the static one of [36] makes all process algebraic operators static – in particular action prefix and choice – so that executed actions and discarded subprocesses are kept within the syntax. There are two separate transition relations, a forward one and a backward one. The two approaches have been shown to be equivalent in terms

of labeled transition system isomorphism [27] and the common properties they exploit to ensure causal reversibility have been systematically classified in [29].

The approach of [36,29] is adequate to study reversibility on basic process calculi and state-transition graphs in the nondeterministic case. Recently we have addressed its adoption in the presence of quantitative information. While it smoothly applies to stochastically timed calculi, for which both causal reversibility and time reversibility [22] have been investigated in [5,4], in the case of deterministically timed calculi its use requires a careful treatment of delays as well as time additivity, laziness, and maximal progress as shown in [6].

In this paper we address reversibility for untimed calculi featuring both nondeterminism and probabilities and show that the approach of [36,29] has to be adapted again, in a way different from [6]. An example of application of such reversible calculi is randomized dining philosophers [31], in which a philosopher may revoke the choice of the first chopstick if the second one is not available (possibly within a short amount of time like in [4]). A different example is given by speculative consumers [39], where to boost parallelism a consumer can probabilistically predict a value on the basis of which to launch a computation, which has to be undone if the guessed value is different from the one sent later by the producer. Yet another example is the smart contract rollback vulnerability [9]; for instance, in a lottery a smart contract makes a probabilistic choice to draft the winning ticket, but an attacker may try to revert the transaction in the case that the purchased ticket is different from the winning one [15].

There are several probabilistic state-transition models that can be used as a basis for our reversible calculus. A limited form of nondeterminism is allowed within reactive models [18], which correspond to Markov decision processes [13] and Rabin probabilistic automata [40]. Like in generative models [18], which correspond to action-labeled discrete-time Markov chains [23], each transition is labeled with an action and an execution probability, but probabilities are enforced only among transitions labeled with the same action. Therefore, in every state a nondeterministic selection is made among transitions labeled with different actions, then a probabilistic selection takes place inside the set of transitions labeled with that action.

Internal nondeterminism, i.e., nondeterministic choices among transitions labeled with the same action, is supported by Segala simple probabilistic automata [42]. In this model every transition is labeled only with an action and goes from a state to a probability distribution over states. In every state the transition to be executed is selected nondeterministically, then the reached state is selected probabilistically among those in the support of the target probability distribution of the chosen transition.

That combination of probability and nondeterminism is called non-alternating to distinguish it from the alternating one of [19]. In the latter model, states are divided into nondeterministic and probabilistic, with transitions being classified as action transitions, which are labeled with an action and go from a nondeterministic state to a probabilistic one, and probabilistic transitions, which are labeled with a probability and go from a probabilistic state to a nondeterministic

one. A more flexible variant, called non-strictly alternating model [35], admits action transitions between two nondeterministic states too.

Both the non-alternating model and the alternating one – whose relationships have been studied in [44] – encompass nondeterministic models, generative models, and reactive models as special cases. Since branching bisimulation semantics plays a fundamental role in reversible systems [12,3,14,15], in this paper we adopt the non-strictly alternating model because in [1] a probabilistic branching bisimulation congruence has been developed for it along with equational and logical characterizations and a polynomial-time decision procedure. In the non-alternating model, for which branching bisimilarity has been just defined in [43], weak variants of bisimulation semantics are more involved because, to achieve transitivity, they require that a single transition be matched by a convex combination of several transitions, corresponding to the use of randomized schedulers; decision procedures can be found in [8,46].

The first contribution of this paper (Section 2) is to show how the general method for reversing process calculi of [36] can be applied to the nondeterministic and probabilistic case based on the non-strictly alternating model. The following adaptations are needed, which are different from those in [6]:

- Similar to executed actions, which have to be decorated with communication keys to know who synchronized with whom when building the backward transition relation [36], probabilistic selections have to be decorated with keys to avoid wrong pairings in the backward direction when they have been performed on both sides of a nondeterministic choice or a parallel composition.
- To comply with the adopted model, in which the forward transitions departing from a state are all either action transitions or probabilistic transitions, like in the forward-only calculus of [1] probabilistic selections have to be made before nondeterministic ones when going forward. As a consequence, a probabilistic selection cannot resolve nondeterministic choices or decide which subprocess advances in a parallel composition. This adds a technical challenge to the definition of the operational semantic rules with respect to [36], as nondeterministic selections – including those among concurrent actions – have to be revoked before probabilistic ones when going backward.

The second contribution (Section 4) is to prove that the resulting calculus meets causal reversibility. This is accomplished through notions of [10] and the technique of [29], which however cannot be applied as they are:

- Conflicting transitions, from which concurrent transitions [10] are then derived, and causal equivalence [10], which is needed to identify computations that differ for the order of concurrent action transitions, have to be extended with additional conditions specific to probabilistic transitions.
- The square property for concurrent transitions, on which the technique of [29] relies to obtain causal reversibility, has to be revised to deal with extended squares that include probabilistic transitions as well.

The paper also features an application of the resulting calculus to a quantum computing example (Section 3) and some directions for future work (Section 5).

$$\begin{array}{|l|}\hline F, G \ ::= \ \underline{0} \mid a \,.\, F \mid F \,{}_p\!\oplus\, G \mid F + G \mid F \parallel_L G \\ R, S \ ::= \ F \mid a[i] \,.\, R \mid R \,{}_{[i]p}\!\oplus\, S \mid R \,{}_p\!\oplus_{[i]}\, S \mid R + S \mid R \parallel_L S \\ \hline \end{array}$$

Table 1. Syntax of forward (top) and reversible (bottom) processes ($. > {}_p\!\oplus > + > \parallel_L$)

2 Reversible Probabilistic Process Calculus

In this section we present the syntax and the semantics of RPPC – Reversible Probabilistic Process Calculus, which is inspired by the process calculi in [20,1] and tailored for a reversible setting according to the static approach of [36].

The syntax of RPPC is shown in Table 1 (along with operator precedence). A standard *forward process* F describes the future behavior and is one of the following: the terminated process $\underline{0}$; the action-prefixed process $a \,.\, F$, which is able to perform action $a \in \mathcal{A}$ and then continues as process F, with action set $\mathcal{A}$ including τ as unobservable action; the probabilistic choice $F \,{}_p\!\oplus\, G$, where F is selected with probability $p \in \mathbb{R}_{]0,1[}$ while G is selected with probability $1-p$; the nondeterministic choice $F + G$, which is resolved based on the actions executable by F and G; or the parallel composition $F \parallel_L G$, where processes F and G execute in parallel and must synchronize only on actions in $L \subseteq \mathcal{A} \setminus \{\tau\}$.

While in [20] there is a strict alternation between nondeterministic processes like $N = \sum_{h \in H} a_h \,.\, P_h$ and probabilistic processes like $P = \bigoplus_{h \in H} \langle p_h \rangle \,.\, N_h$, as in [1] we stipulate for our more liberal syntax that probabilistic choices have to be resolved *before* nondeterministic ones when going forward, so that every process either executes actions or makes probabilistic selections and no consecutive probabilistic transitions are possible. Thus, similar to time determinism in [6], a probabilistic selection cannot resolve nondeterministic choices or decide who advances in a parallel composition. When each subprocess of a nondeterministic choice or a parallel composition makes a probabilistic selection, the corresponding probabilities are then multiplied at the level of the entire process.

A *reversible process* R includes the past behavior. The syntax of reversible processes differs from the one of forward processes due to the fact that, in the former, actions and probabilities may be decorated. As in [36], an action is decorated with a *communication key* i belonging to a countable set $\mathcal{K}$. A process of the form $a[i] \,.\, R$ expresses that in the past it synchronized with the environment on a and this synchronization was identified by key i. Keys are thus attached only to executed actions and, as we will see, are necessary to remember who synchronized with whom when undoing actions; keys could be omitted in the absence of parallel composition. Processes $R \,{}_{[i]p}\!\oplus\, S$ and $R \,{}_p\!\oplus_{[i]}\, S$ indicate that in the past a probabilistic selection was made in favor of the left or the right subprocess, respectively. We will see that communication keys are needed to avoid wrong pairings of probabilistic selections in the backward direction; keys could be omitted in the absence of nondeterministic choice and parallel composition.

We denote by $\mathcal{P}$ the set of processes generated by the productions for R in Table 1, while we use predicate $\mathtt{std}(\_)$ to identify the standard forward processes that can be derived from the productions for F in the same table. For example,

$a\,.\,(b\,.\,\underline{0}\,_{0.5}\!\oplus\, c\,.\,\underline{0})$ is a standard forward process that can execute action a and then probabilistically selects between doing action b or doing action c, while $a[i]\,.\,(b\,.\,\underline{0}\,_{[j]0.5}\!\oplus\, c\,.\,\underline{0})$ is a reversible process that can either undo the probabilistic selection in favor of b (key j) and then action a (key i), or perform action b. Note that $a\,.\,(b\,.\,\underline{0}\,_{[j]0.5}\!\oplus\, c\,.\,\underline{0})$ and $a\,.\,(b[i]\,.\,\underline{0}\,_{0.5}\!\oplus\, c\,.\,\underline{0})$ are not in $\mathcal{P}$ as a future action or probabilistic selection cannot precede a past one in the description of a process.

Let $\mathcal{A}_\mathcal{K} = \mathcal{A} \times \mathcal{K}$ and $\mathbb{R}_\mathcal{K} = \mathbb{R}_{]0,1[} \times \mathcal{K}$, with $\mathcal{L} = \mathcal{A}_\mathcal{K} \cup \mathbb{R}_\mathcal{K}$ ranged over by ℓ. The semantics for RPPC is the labeled transition system $(\mathcal{P}, \mathcal{L}, \longmapsto)$. The transition relation $\longmapsto \subseteq \mathcal{P} \times \mathcal{L} \times \mathcal{P}$ is given by $\longmapsto = \longrightarrow \cup \dashrightarrow$ where in turn the *forward transition relation* is given by $\longrightarrow = \longrightarrow_{\mathtt{a}} \cup \longrightarrow_{\mathtt{p}}$ and the *backward transition relation* is given by $\dashrightarrow = \dashrightarrow_{\mathtt{a}} \cup \dashrightarrow_{\mathtt{p}}$. In the definitions of the transition relations, we make use of the set $\mathtt{key_a}(R)$ of action keys and of the set $\mathtt{key_p}(R)$ of probabilistic selection keys occurring in $R \in \mathcal{P}$:

$$\begin{array}{rcl@{\qquad}rcl}
\mathtt{key_a}(F) & = & \emptyset & \mathtt{key_p}(F) & = & \emptyset \\
\mathtt{key_a}(a[i]\,.\,R) & = & \{i\} \cup \mathtt{key_a}(R) & \mathtt{key_p}(a[i]\,.\,R) & = & \mathtt{key_p}(R) \\
\mathtt{key_a}(R\,_{[i]p}\!\oplus\, S) & = & \mathtt{key_a}(R) & \mathtt{key_p}(R\,_{[i]p}\!\oplus\, S) & = & \{i\} \cup \mathtt{key_p}(R) \\
\mathtt{key_a}(R\,_{p}\!\oplus_{[i]}\, S) & = & \mathtt{key_a}(S) & \mathtt{key_p}(R\,_{p}\!\oplus_{[i]}\, S) & = & \{i\} \cup \mathtt{key_p}(S) \\
\mathtt{key_a}(R + S) & = & \mathtt{key_a}(R) \cup \mathtt{key_a}(S) & \mathtt{key_p}(R + S) & = & \mathtt{key_p}(R) \cup \mathtt{key_p}(S) \\
\mathtt{key_a}(R \parallel_L S) & = & \mathtt{key_a}(R) \cup \mathtt{key_a}(S) & \mathtt{key_p}(R \parallel_L S) & = & \mathtt{key_p}(R) \cup \mathtt{key_p}(S)
\end{array}$$

as well as of predicate $\mathtt{npa}(\_)$ to establish whether the considered process $R \in \mathcal{P}$ contains no past actions (note that $\mathtt{std}(R)$ ensures $\mathtt{npa}(R)$):

$$\begin{array}{rcl@{\qquad}rcl}
\mathtt{npa}(F) & = & \mathtt{true} & \mathtt{npa}(a[i]\,.\,R) & = & \mathtt{false} \\
\mathtt{npa}(R\,_{[i]p}\!\oplus\, S) & = & \mathtt{npa}(R) & \mathtt{npa}(R\,_{p}\!\oplus_{[i]}\, S) & = & \mathtt{npa}(S) \\
\mathtt{npa}(R + S) & = & \mathtt{npa}(R) \wedge \mathtt{npa}(S) & \mathtt{npa}(R \parallel_L S) & = & \mathtt{npa}(R) \wedge \mathtt{npa}(S)
\end{array}$$

The *action transition relations* $\longrightarrow_{\mathtt{a}} \subseteq \mathcal{P} \times \mathcal{A}_\mathcal{K} \times \mathcal{P}$ and $\dashrightarrow_{\mathtt{a}} \subseteq \mathcal{P} \times \mathcal{A}_\mathcal{K} \times \mathcal{P}$ are the least relations respectively induced by the forward rules in the left part of Table 2 and by the backward rules in the right part of the same table.

Rule Act1 handles processes of the form $a\,.\,F$, where F is written as R subject to $\mathtt{std}(R)$. In addition to transforming the action prefix into a transition label, it generates a key i that is bound to action a thus yielding the label $a[i]$. As can be noted, according to [36] the prefix is not discarded by the application of the rule, instead it becomes a key-storing part of the target process that is necessary to offer again that action after rolling back. Rule Act1$^\bullet$ reverts action $a[i]$ of process $a[i]\,.\,R$ provided that R is a standard process, which ensures that $a[i]$ is the only executed action that is left to undo.

The presence of rules Act2 and Act2$^\bullet$ is motivated by the fact that rule Act1 does not discard the executed action from the process it generates. In particular, rule Act2 allows a process $a[i]\,.\,R$ to execute if R itself can execute, provided that the action performed by R picks a key j different from i so that all the action prefixes in a sequence are decorated with distinct keys. Rule Act2$^\bullet$ simply propagates the execution of backward actions from inner subprocesses that are not standard by preserving key uniqueness, in such a way that executed actions are undone from the most recent one to the least recent one.

Rules Act3 and Act3$^\bullet$, along with their omitted symmetric variants for $R\,_{p}\!\oplus_{[i]}\, S$ (in which S has been selected with probability $1-p$), play an analogous

$$(\textsc{Act1})\ \frac{\mathtt{std}(R)}{a\,.\,R \xrightarrow{a[i]}_{\mathtt{a}} a[i]\,.\,R} \qquad (\textsc{Act1}^\bullet)\ \frac{\mathtt{std}(R)}{a[i]\,.\,R \dashrightarrow_{\mathtt{a}}^{a[i]} a\,.\,R}$$

$$(\textsc{Act2})\ \frac{R \xrightarrow{b[j]}_{\mathtt{a}} R' \quad j \neq i}{a[i]\,.\,R \xrightarrow{b[j]}_{\mathtt{a}} a[i]\,.\,R'} \qquad (\textsc{Act2}^\bullet)\ \frac{R \dashrightarrow_{\mathtt{a}}^{b[j]} R' \quad j \neq i}{a[i]\,.\,R \dashrightarrow_{\mathtt{a}}^{b[j]} a[i]\,.\,R'}$$

$$(\textsc{Act3})\ \frac{R \xrightarrow{b[j]}_{\mathtt{a}} R'}{R \,{}_{[i]p}\!\oplus S \xrightarrow{b[j]}_{\mathtt{a}} R' \,{}_{[i]p}\!\oplus S} \qquad (\textsc{Act3}^\bullet)\ \frac{R \dashrightarrow_{\mathtt{a}}^{b[j]} R'}{R \,{}_{[i]p}\!\oplus S \dashrightarrow_{\mathtt{a}}^{b[j]} R' \,{}_{[i]p}\!\oplus S}$$

$$(\textsc{Cho})\ \frac{R \xrightarrow{a[i]}_{\mathtt{a}} R' \quad \mathtt{npa}(S) \quad S \not\rightarrow_{\mathtt{p}}}{R + S \xrightarrow{a[i]}_{\mathtt{a}} R' + S} \qquad (\textsc{Cho}^\bullet)\ \frac{R \dashrightarrow_{\mathtt{a}}^{a[i]} R' \quad \mathtt{npa}(S) \quad S \not\rightarrow_{\mathtt{p}}}{R + S \dashrightarrow_{\mathtt{a}}^{a[i]} R' + S}$$

$$(\textsc{Par})\ \frac{R \xrightarrow{a[i]}_{\mathtt{a}} R' \quad a \notin L \quad i \notin \mathtt{key}_{\mathtt{a}}(S) \quad S \not\rightarrow_{\mathtt{p}}}{R \parallel_L S \xrightarrow{a[i]}_{\mathtt{a}} R' \parallel_L S} \qquad (\textsc{Par}^\bullet)\ \frac{R \dashrightarrow_{\mathtt{a}}^{a[i]} R' \quad a \notin L \quad i \notin \mathtt{key}_{\mathtt{a}}(S) \quad S \not\rightarrow_{\mathtt{p}}}{R \parallel_L S \dashrightarrow_{\mathtt{a}}^{a[i]} R' \parallel_L S}$$

$$(\textsc{Coo})\ \frac{R \xrightarrow{a[i]}_{\mathtt{a}} R' \quad S \xrightarrow{a[i]}_{\mathtt{a}} S' \quad a \in L}{R \parallel_L S \xrightarrow{a[i]}_{\mathtt{a}} R' \parallel_L S'} \qquad (\textsc{Coo}^\bullet)\ \frac{R \dashrightarrow_{\mathtt{a}}^{a[i]} R' \quad S \dashrightarrow_{\mathtt{a}}^{a[i]} S' \quad a \in L}{R \parallel_L S \dashrightarrow_{\mathtt{a}}^{a[i]} R' \parallel_L S'}$$

Table 2. Operational semantic rules for RPPC action transitions

propagating role for a resolved probabilistic choice. Note that executed actions and resolved probabilistic choices are not required to feature different keys.

Unlike the classical rules for nondeterministic choice [33], according to [36] rule Cho does not discard the part of the overall process that has not contributed to the executed action. If process R does an action, say $a[i]$, and becomes R', then the entire process $R + S$ becomes $R' + S$ as the information about $+\,S$, where S contains no past actions, is necessary for offering again the original choice after rolling back. Once the choice is made, only the non-standard process R' can proceed further because process S – which is standard or contains resolved probabilistic choices – constitutes a dead context of R'. Moreover, since we have stipulated that probabilistic choices have to be resolved before nondeterministic ones when going forward, $R + S$ can perform $a[i]$ and become $R' + S$ if S has no probabilistic transitions, which is denoted by $S \not\rightarrow_{\mathtt{p}}$. Rule Cho$^\bullet$ has precisely the same structure as rule Cho, but deals with the backward transition relation; if R' is standard, then the dead context S will come into play again. The symmetric variants of Cho and Cho$^\bullet$, in which it is S to move, are omitted. Note that, in order to apply Cho (resp. Cho$^\bullet$) or its symmetric variant, at least one of R and S must contain no past actions, meaning that it is impossible for two processes containing past actions to execute if they are composed by a choice operator.

The semantics of parallel composition is inspired by [21]. Rule Par allows process R within $R \parallel_L S$ to individually perform an action $a[i]$ provided $a \notin L$. It is also checked that the executing action is bound to a key $i \notin \mathtt{key}_{\mathtt{a}}(S)$,

thus ensuring the uniqueness of communication keys across parallel composition too. Moreover, since we have stipulated that, when going forward, probabilistic choices have to be resolved before nondeterministic ones (including those arising from action interleaving), it is further verified that S has no probabilistic transitions. Rule PAR$^\bullet$ has the same structure as PAR; their symmetric variants are omitted. Rules COO and COO$^\bullet$ instead allow both R and S to move by synchronizing on any action in the set L as long as the communication key is the same on both sides. The resulting cooperation action has the same name and the same key as the two synchronizing actions.

To illustrate the need of communication keys [36], consider the standard forward process $(a \,.\, F_1 \parallel_\emptyset a \,.\, F_2) \parallel_{\{a\}} (a \,.\, F_3 \parallel_\emptyset a \,.\, F_4)$, which may evolve to either the reversible process $(a[i] \,.\, F_1 \parallel_\emptyset a[j] \,.\, F_2) \parallel_{\{a\}} (a[i] \,.\, F_3 \parallel_\emptyset a[j] \,.\, F_4)$ or the reversible process $(a[i] \,.\, F_1 \parallel_\emptyset a[j] \,.\, F_2) \parallel_{\{a\}} (a[j] \,.\, F_3 \parallel_\emptyset a[i] \,.\, F_4)$ after performing a forward $a[i]$-transition followed by a forward $a[j]$-transition. When going backward, in the absence of the two distinct communication keys i and j we do not know that the a preceding F_1 (resp. F_2) synchronized with the a preceding F_3 (resp. F_4) in the first case or the a preceding F_4 (resp. F_3) in the second case.

The *probabilistic transition relations* $\longrightarrow_{\mathtt{p}} \subseteq \mathcal{P} \times \mathbb{R}_{\mathcal{K}} \times \mathcal{P}$ and $\dashrightarrow_{\mathtt{p}} \subseteq \mathcal{P} \times \mathbb{R}_{\mathcal{K}} \times \mathcal{P}$ are the least relations respectively induced by the forward rules in the left part of Table 3 and by the backward rules in the right part of the same table. Each backward probabilistic transition is conventionally labeled with the same probability as the corresponding forward transition; note however that this probabilistic value is meaningful only in the forward direction.

Rules PSEL1 and PSEL2 handle probabilistic selections between standard processes. The former rule describes the case in which R has no probabilistic choices, hence the probability of selecting R is simply p. The latter rule describes the case in which R has probabilistic choices, so that p is multiplied by the probability of selecting R'. To enable reversibility, in both rules the probability associated with the operator is decorated with a unique key and the subprocess that has not been selected is not discarded. Rules PSEL1$^\bullet$ and PSEL2$^\bullet$ are the backward counterparts. The symmetric variants for $R \,{}_p\!\oplus_{[i]}\, S$ (in which S has been selected with probability $1-p$) are omitted. For processes like $\underline{0} \,{}_{0.5}\!\oplus\, \underline{0}$ two distinct transitions $\underline{0} \,{}_{0.5}\!\oplus\, \underline{0} \xrightarrow{(0.5)^{[i]}}_{\mathtt{p}} \underline{0} \,{}_{[i]0.5}\!\oplus\, \underline{0}$ and $\underline{0} \,{}_{0.5}\!\oplus\, \underline{0} \xrightarrow{(0.5)^{[i]}}_{\mathtt{p}} \underline{0} \,{}_{0.5}\!\oplus_{[i]}\, \underline{0}$ are generated thanks to decorations in distinct positions within the two target processes, thus avoiding to resort to multisets of probabilistic transitions [20].

Rules PSEL3 and PSEL3$^\bullet$ propagate probabilistic selections in the context of resolved probabilistic choices followed by executed actions; their symmetric variants are omitted. Rules PSEL4 and PSEL4$^\bullet$ propagate probabilistic selections in the context of executed actions only. We remind that executed actions and resolved probabilistic choices are not required to feature different keys.

Rule PCHO1 represents a probabilistic selection made within a nondeterministic choice by R alone as $S \not\dashrightarrow_{\mathtt{p}}$, provided that S contains no past actions and key i does not occur in S, with PCHO1$^\bullet$ being its backward counterpart. Their symmetric variants are omitted. Note that, as a consequence of the fact that probabilistic choices have to be resolved before nondeterministic choices when

$$(\text{PSEL1})\ \frac{\mathtt{std}(R) \quad \mathtt{std}(S) \quad R \not\xrightarrow{}_{\mathtt{p}}}{R\,{}_p\!\oplus S \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R\,{}_{[i]p}\!\oplus S} \qquad (\text{PSEL1}^\bullet)\ \frac{\mathtt{std}(R) \quad \mathtt{std}(S) \quad R \not\xrightarrow{}_{\mathtt{p}}}{R\,{}_{[i]p}\!\oplus S \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R\,{}_p\!\oplus S}$$

$$(\text{PSEL2})\ \frac{R \xrightarrow{(q)^{[j]}}_{\mathtt{p}} R' \quad \mathtt{std}(R) \quad \mathtt{std}(S) \quad i \notin \mathtt{key}_{\mathtt{p}}(R')}{R\,{}_p\!\oplus S \xrightarrow{(p\cdot q)^{[i]}}_{\mathtt{p}} R'\,{}_{[i]p}\!\oplus S} \qquad (\text{PSEL2}^\bullet)\ \frac{R \xrightarrow{\quad(q)^{[j]}\quad}_{\mathtt{p}} R' \quad \mathtt{std}(R') \quad \mathtt{std}(S) \quad i \notin \mathtt{key}_{\mathtt{p}}(R)}{R\,{}_{[i]p}\!\oplus S \xrightarrow{\quad(p\cdot q)^{[i]}\quad}_{\mathtt{p}} R'\,{}_p\!\oplus S}$$

$$(\text{PSEL3})\ \frac{R \xrightarrow{(q)^{[j]}}_{\mathtt{p}} R' \quad \neg\mathtt{std}(R) \quad j \neq i}{R\,{}_{[i]p}\!\oplus S \xrightarrow{(q)^{[j]}}_{\mathtt{p}} R'\,{}_{[i]p}\!\oplus S} \qquad (\text{PSEL3}^\bullet)\ \frac{R \xrightarrow{\quad(q)^{[j]}\quad}_{\mathtt{p}} R' \quad \neg\mathtt{std}(R') \quad j \neq i}{R\,{}_{[i]p}\!\oplus S \xrightarrow{\quad(q)^{[j]}\quad}_{\mathtt{p}} R'\,{}_{[i]p}\!\oplus S}$$

$$(\text{PSEL4})\ \frac{R \xrightarrow{(q)^{[j]}}_{\mathtt{p}} R'}{a[i]\,.\,R \xrightarrow{(q)^{[j]}}_{\mathtt{p}} a[i]\,.\,R'} \qquad (\text{PSEL4}^\bullet)\ \frac{R \xrightarrow{\quad(q)^{[j]}\quad}_{\mathtt{p}} R'}{a[i]\,.\,R \xrightarrow{\quad(q)^{[j]}\quad}_{\mathtt{p}} a[i]\,.\,R'}$$

$$(\text{PCHO1})\ \frac{R \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' \quad i \notin \mathtt{key}_{\mathtt{p}}(S) \quad \mathtt{npa}(S) \quad S \not\xrightarrow{}_{\mathtt{p}}}{R + S \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' + S} \qquad (\text{PCHO1}^\bullet)\ \frac{R \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R' \quad i \notin \mathtt{key}_{\mathtt{p}}(S) \quad \mathtt{npa}(S) \quad S \not\xrightarrow{}_{\mathtt{p}}}{R + S \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R' + S}$$

$$(\text{PCHO2})\ \frac{R \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' \quad S \xrightarrow{(q)^{[i]}}_{\mathtt{p}} S'}{R + S \xrightarrow{(p\cdot q)^{[i]}}_{\mathtt{p}} R' + S'} \qquad (\text{PCHO2}^\bullet)\ \frac{R \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R' \quad S \xrightarrow{\quad(q)^{[i]}\quad}_{\mathtt{p}} S'}{R + S \xrightarrow{\quad(p\cdot q)^{[i]}\quad}_{\mathtt{p}} R' + S'}$$

$$(\text{PPAR})\ \frac{R \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' \quad i \notin \mathtt{key}_{\mathtt{p}}(S) \quad S \not\xrightarrow{}_{\mathtt{p}}}{R \parallel_L S \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' \parallel_L S} \qquad (\text{PPAR}^\bullet)\ \frac{R \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R' \quad i \notin \mathtt{key}_{\mathtt{p}}(S) \quad S \not\xrightarrow{}_{\mathtt{p}} \quad \mathtt{npa}(S) \vee \neg\mathtt{npa}(R)}{R \parallel_L S \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R' \parallel_L S}$$

$$(\text{PCOO})\ \frac{R \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' \quad S \xrightarrow{(q)^{[i]}}_{\mathtt{p}} S'}{R \parallel_L S \xrightarrow{(p\cdot q)^{[i]}}_{\mathtt{p}} R' \parallel_L S'} \qquad (\text{PCOO}^\bullet)\ \frac{R \xrightarrow{\quad(p)^{[i]}\quad}_{\mathtt{p}} R' \quad S \xrightarrow{\quad(q)^{[i]}\quad}_{\mathtt{p}} S'}{R \parallel_L S \xrightarrow{\quad(p\cdot q)^{[i]}\quad}_{\mathtt{p}} R' \parallel_L S'}$$

Table 3. Operational semantic rules for RPPC probabilistic transitions

going forward, nondeterministic choices have to be revoked before probabilistic choices when going backward. For instance, $(a\,.\,\underline{0}\,{}_p\!\oplus\, b\,.\,\underline{0}) + c\,.\,\underline{0}$ first resolves the probabilistic choice thus becoming e.g. $(a\,.\,\underline{0}\,{}_{[i]p}\!\oplus\, b\,.\,\underline{0}) + c\,.\,\underline{0}$ and then can perform e.g. c thus evolving to $(a\,.\,\underline{0}\,{}_{[i]p}\!\oplus\, b\,.\,\underline{0}) + c[j]\,.\,\underline{0}$, where the probabilistic choice cannot be revoked – otherwise $(a\,.\,\underline{0}\,{}_p\!\oplus\, b\,.\,\underline{0}) + c[j]\,.\,\underline{0}$ would be reached that cannot be encountered when going forward – thanks to the `npa` constraint.

Rule PCHO2 expresses instead the fact that, since a probabilistic selection cannot decide which subprocess is chosen in a nondeterministic choice, if each subprocess of a nondeterministic choice makes a probabilistic selection then the two selections are synchronized and the corresponding probabilities are multiplied. Rule PCHO2$^\bullet$ plays the corresponding role in the backward direction.

Likewise, rule PPar represents a probabilistic selection made within a parallel composition by R alone as $S \not\xrightarrow{\mathtt{p}}$, provided that key i does not occur in S, with PPar$^\bullet$ being its backward counterpart. Their symmetric variants are omitted. Note that PPar$^\bullet$ additionally requires $\mathtt{npa}(S) \vee \neg\mathtt{npa}(R)$. The $\mathtt{npa}(S)$ part stems from the fact that nondeterministic choices, including those between two interleaving actions in a parallel composition, have to be revoked before probabilistic choices when going backward. As an example, $(a\,.\,\underline{0}\,{}_p\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset c\,.\,\underline{0}$ first resolves the probabilistic choice thus becoming e.g. $(a\,.\,\underline{0}\,{}_{[i]p}\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset c\,.\,\underline{0}$ and then can perform e.g. c thus evolving to $(a\,.\,\underline{0}\,{}_{[i]p}\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset c[j]\,.\,\underline{0}$, where the probabilistic choice cannot be revoked – otherwise $(a\,.\,\underline{0}\,{}_p\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset c[j]\,.\,\underline{0}$ would be reached that cannot be encountered when going forward – thanks to the additional constraint. The $\neg\mathtt{npa}(R)$ part is needed when two probabilistic choices are preceded by two interleaving actions like in $a_1\,.\,(b\,.\,\underline{0}\,{}_p\!\oplus\, c\,.\,\underline{0})\parallel_\emptyset a_2\,.\,(d\,.\,\underline{0}\,{}_q\!\oplus\, e\,.\,\underline{0})$. While going forward this can reach e.g. $a_1[i_1]\,.\,(b\,.\,\underline{0}\,{}_{[j_1]p}\!\oplus\, c\,.\,\underline{0})\parallel_\emptyset a_2[i_2]\,.\,(d\,.\,\underline{0}\,{}_q\!\oplus_{[j_2]}\, e\,.\,\underline{0})$, from which it must be possible to revoke either probabilistic choice.

Rule PCoo represents instead the fact that, since a probabilistic selection cannot decide which subprocess advances in a parallel composition, if each subprocess of a parallel composition makes a probabilistic selection then the two selections are synchronized and the corresponding probabilities are multiplied. Rule PCoo$^\bullet$ plays the corresponding role in the backward direction.

To illustrate the need of communication keys also for probabilistic choices (not in [36]), consider the standard forward process $(a\,.\,\underline{0}\,{}_p\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset (c\,.\,\underline{0}\,{}_q\!\oplus\, d\,.\,\underline{0})$, which may evolve to one of the following four reversible processes:

- $(a\,.\,\underline{0}\,{}_{[i]p}\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset (c\,.\,\underline{0}\,{}_{[i]q}\!\oplus\, d\,.\,\underline{0})$ with probability $p\cdot q$.
- $(a\,.\,\underline{0}\,{}_{[i]p}\!\oplus\, b\,.\,\underline{0})\parallel_\emptyset (c\,.\,\underline{0}\,{}_q\!\oplus_{[i]}\, d\,.\,\underline{0})$ with probability $p\cdot(1-q)$.
- $(a\,.\,\underline{0}\,{}_p\!\oplus_{[i]}\, b\,.\,\underline{0})\parallel_\emptyset (c\,.\,\underline{0}\,{}_{[i]q}\!\oplus\, d\,.\,\underline{0})$ with probability $(1-p)\cdot q$.
- $(a\,.\,\underline{0}\,{}_p\!\oplus_{[i]}\, b\,.\,\underline{0})\parallel_\emptyset (c\,.\,\underline{0}\,{}_q\!\oplus_{[i]}\, d\,.\,\underline{0})$ with probability $(1-p)\cdot(1-q)$.

When going backward, in the absence of communication key i we do not know which subprocess of the probabilistic choice on the left of $\parallel_\emptyset$ was combined with which subprocess of the probabilistic choice on the right of $\parallel_\emptyset$.

It may be argued that what really matters is the position of the key with respect to the probabilistic parameter, hence a uniform decoration within every occurrence of the probabilistic choice operator – e.g., ${}_{\langle p}\!\oplus$ and ${}_{p\rangle}\!\oplus$ – would suffice. However, consider the standard forward process $a\,.\,(b\,.\,\underline{0}\,{}_p\!\oplus\, c\,.\,\underline{0})\ \#\ (d\,.\,\underline{0}\,{}_q\!\oplus\, e\,.\,\underline{0})$ where $\#\in\{+,\parallel_\emptyset\}$. The only initial option is resolving the probabilistic choice on the right, thus reaching for instance $a\,.\,(b\,.\,\underline{0}\,{}_p\!\oplus\, c\,.\,\underline{0})\ \#\ (d\,.\,\underline{0}\,{}_{[i]q}\!\oplus\, e\,.\,\underline{0})$. Then suppose that a is executed, which can only be followed by resolving the probabilistic choice on the left, which yields for instance $a[j]\,.\,(b\,.\,\underline{0}\,{}_p\!\oplus_{[k]}\, c\,.\,\underline{0})\ \#\ (d\,.\,\underline{0}\,{}_{[i]q}\!\oplus\, e\,.\,\underline{0})$. Now, in the backward direction, the two probabilistic selections cannot be undone together – thus reaching the inconsistent $a[j]\,.\,(b\,.\,\underline{0}\,{}_p\!\oplus\, c\,.\,\underline{0})\ \#\ (d\,.\,\underline{0}\,{}_q\!\oplus\, e\,.\,\underline{0})$ – because $i\neq k$ and hence PCho2$^\bullet$ and PCoo$^\bullet$ are not applicable.

Process syntax prevents future actions or probabilistic selections from preceding past ones as well as both sides of a probabilistic choice from being simultaneously selected. However, these are not the only necessary limitations,

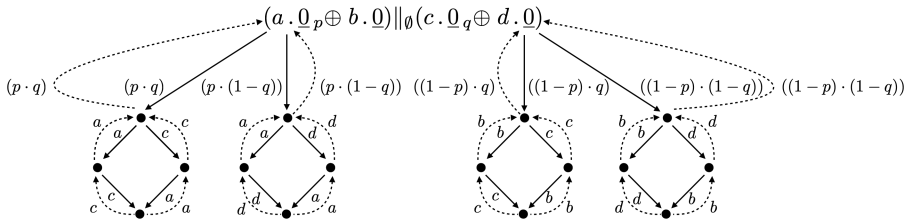

Fig. 1. Labeled transition system underlying $(a\,.\,\underline{0}\,{}_p\oplus\, b\,.\,\underline{0})\,\|_\emptyset(c\,.\,\underline{0}\,{}_q\oplus\, d\,.\,\underline{0})$

because not all the processes generated by the considered grammar are semantically meaningful. In the case of a nondeterministic choice at least one of the two subprocesses has to contain no past actions (but can contain resolved probabilistic choices), hence for instance $a[i]\,.\,\underline{0} + b[j]\,.\,\underline{0}$ is not admissible as it indicates that both branches have been selected. Moreover, key uniqueness must be enforced within processes featuring executed actions or resolved probabilistic choices, so for example $a[i]\,.\,b[i]\,.\,\underline{0}$, $a[i]\,.\,\underline{0}\,\|_\emptyset\, b[i]\,.\,\underline{0}$, $F_1\,{}_{[i]p}\oplus\, F_2\,{}_{[i]q}\oplus\, F_3$, and $\underline{0}\,{}_p\oplus_{[i]}\, a[j]\,.\,F\,\#\,a[j]\,.\,(F_1\,{}_{[i]q}\oplus\, F_2)$ where $\#\in\{+,\|_{\{a\}}\}$ are not admissible either.

In the following we thus consider only *reachable processes*, whose set we denote by $\mathbb{P}$. They include processes from which a computation can start, i.e., standard forward processes, as well as processes that can be derived from the previous ones via finitely many applications of the semantic rules. Given a reachable process $R\in\mathbb{P}$, if $\mathtt{npa}(R)$ then $\mathtt{key}_\mathtt{a}(R)=\emptyset$ while $\mathtt{key}_\mathtt{a}(R')\neq\emptyset$ for any other process R' reachable from R in which at least one of the actions occurring in R has been executed, as that action has been equipped with a key inside R'.

We conclude by discussing some properties of the resulting labeled transition system $(\mathbb{P},\mathcal{L},\longmapsto)$, where we recall that $\longmapsto\;=\;\longrightarrow\cup\;\dashrightarrow$, $\longrightarrow\;=\;\longrightarrow_\mathtt{a}\cup\longrightarrow_\mathtt{p}$, and $\dashrightarrow\;=\;\dashrightarrow_\mathtt{a}\cup\dashrightarrow_\mathtt{p}$. First of all, we observe that forward probabilistic transitions across a parallel composition are combined with each other into single transitions by rule PCoo in the same way as nested probabilistic choices yield a single forward probabilistic transition by rule PSel2 and its symmetric variant. Therefore, every state reached by a forward probabilistic transition cannot have forward probabilistic transitions and hence the labeled transition system cannot feature squares composed of forward probabilistic transitions, whereas it can contain squares made out of forward action transitions due to the interleaving of concurrent actions. As an example consider again $(a\,.\,\underline{0}\oplus_p b\,.\,\underline{0})\,\|_\emptyset(c\,.\,\underline{0}\oplus_q d\,.\,\underline{0})$, whose underlying labeled transition system is depicted in Figure 1 up to keys.

Secondly, when focusing only on the forward transition relation $\longrightarrow\;=\;\longrightarrow_\mathtt{a}\cup\longrightarrow_\mathtt{p}$, the labeled transition system is consistent with the non-strict variant [35] of the alternating model [20]. This means that in every state all the forward transitions are either action transitions, in which case the state is nondeterministic, or probabilistic transitions, in which case the state is probabilistic. The alternation is not strict because, while a probabilistic state can reach via forward transitions only nondeterministic states due to rule PSel2 and its symmetric variant as well as rule PCoo, a nondeterministic state can reach via forward transitions either

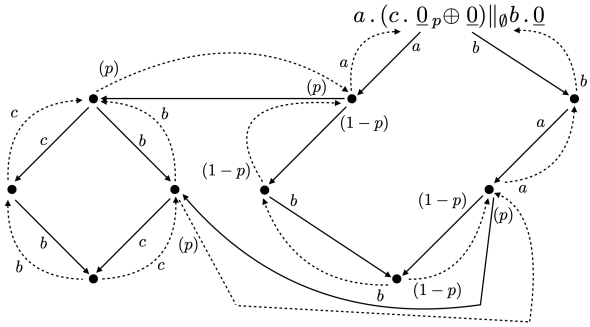

Fig. 2. Labeled transition system underlying $a\,.\,(c\,.\,\underline{0}\,{}_p\!\oplus\,\underline{0})\,\|_\emptyset\,b\,.\,\underline{0}$

probabilistic states or other nondeterministic states. In contrast, a state can have both backward action transitions and backward probabilistic transitions. This may happen when transitions arising from probabilistic selections are involved in squares along with action transitions stemming from the interleaving of concurrent actions. For instance consider $a\,.\,(c\,.\,\underline{0}\,{}_p\!\oplus\,\underline{0})\,\|_\emptyset\,b\,.\,\underline{0}$, whose underlying labeled transition system is depicted in Figure 2, and look at the rectangle with action and probabilistic transitions on the right as well as its bottommost state.

3 Example: Applying RPPC to Quantum Computing

A quantum bit, or qubit, is a physical system with two basis states conventionally denoted by $|0\rangle$ and $|1\rangle$, which correspond to one-bit classical values. According to quantum theory, a general state of a quantum system is a superposition or linear combination of basis states. For instance, a qubit has state $\alpha|0\rangle + \beta|1\rangle$ where $\alpha, \beta \in \mathbb{C}$ satisfy $|\alpha|^2 + |\beta|^2 = 1$. In RPPC we can model a qubit as:

$$Q \;=\; m\,.\,(z\,{}_p\!\oplus\,o)$$

where m stands for measurement, z for zero, and o for one. Measuring its value destroys superposition and yields 0 with probability $p = |\alpha|^2$ and 1 with probability $1 - p = |\beta|^2$, leaving the system in state $|0\rangle$ or $|1\rangle$ respectively.

Tensor product is used to represent systems made out of several qubits. For example, a 2-qubit system has basis states $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$. Its general state is $\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$ where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$. Measuring the first (resp. second) qubit only yields 0 with probability $|\alpha|^2 + |\beta|^2$ (resp. $|\alpha|^2 + |\gamma|^2$) and 1 with probability $|\gamma|^2 + |\delta|^2$ (resp. $|\beta|^2 + |\delta|^2$), leaving the system in state $\frac{1}{\sqrt{|\alpha|^2+|\beta|^2}}(\alpha|00\rangle + \beta|01\rangle)$ or state $\frac{1}{\sqrt{|\gamma|^2+|\delta|^2}}(\gamma|10\rangle + \delta|11\rangle)$ (resp. in state $\frac{1}{\sqrt{|\alpha|^2+|\gamma|^2}}(\alpha|00\rangle + \gamma|10\rangle)$ or state $\frac{1}{\sqrt{|\beta|^2+|\delta|^2}}(\beta|01\rangle + \delta|11\rangle)$) respectively.

If both qubits are measured simultaneously, then the possible results are 0, 1, 2, 3 with corresponding probabilities $|\alpha|^2$, $|\beta|^2$, $|\gamma|^2$, $|\delta|^2$ and states $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$, respectively. This can be modeled in RPPC as follows:

$$QQ \;=\; m\,.\,(z\,.\,(z\,{}_{q_1}\!\oplus\,o)\,{}_p\!\oplus\,o\,.\,(z\,{}_{q_2}\!\oplus\,o))$$

where $p \cdot q_1 = |\alpha|^2$, $p \cdot (1 - q_1) = |\beta|^2$, $(1 - p) \cdot q_2 = |\gamma|^2$, $(1 - p) \cdot (1 - q_2) = |\delta|^2$.

In a closed quantum system, all the operations on qubits are reversible. A common example of such an operation is the quantum controlled-NOT gate, or CNOT gate. This gate acts on two qubits, often called the control qubit and the target qubit. The CNOT gate flips the state of the target qubit if and only if the control qubit is in state $|1\rangle$. It is reversible because it is a unitary operation, meaning that it can be undone by applying the same operation again, like NOT on classical bits. The truth table of CNOT is as follows:

control input	target input	control output	target output
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

We can model a CNOT gate in RPPC as follows, where we use primes to distinguish output bits from input ones:

$$CNOT = m\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o\,.\,z\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z')$$

and then apply it to the 2-qubit system as follows:

$$QQ \parallel_L CNOT$$

where $L = \{m, z, o\}$. The following execution takes place when the input is $|10\rangle$ – o and z are executed – in which case $|11\rangle$ is returned – o' and o' are executed:

$$\begin{array}{rl}
\xrightarrow{m[1]}_{\mathtt{a}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p} \oplus o\,.\,(z_{\,q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o\,.\,z\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z') \\
\xrightarrow{(1-p)^{[2]}}_{\mathtt{p}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o\,.\,(z_{\,q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o\,.\,z\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z') \\
\xrightarrow{o[3]}_{\mathtt{a}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o[3]\,.\,(z_{\,q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o[3]\,.\,z\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z') \\
\xrightarrow{(q_2)^{[4]}}_{\mathtt{p}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o[3]\,.\,(z_{\,[4]q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o[3]\,.\,z\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z') \\
\xrightarrow{z[5]}_{\mathtt{a}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o[3]\,.\,(z[5]_{\,q_2[4]} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o[3]\,.\,z[5]\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z') \\
\xrightarrow{o'[6]}_{\mathtt{a}}\xrightarrow{o'[7]}_{\mathtt{a}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o[3]\,.\,(z[5]_{\,q_2[4]} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o[3]\,.\,z[5]\,.\,o'[6]\,.\,o'[7] + o\,.\,o\,.\,o'.\,z')
\end{array}$$

where action and selection keys are all distinct for the sake of readability.

Note that when performing $o[3]$ process $CNOT$ may wrongly select the fourth branch of its nondeterministic choice, i.e., $o\,.\,o\,.\,o'.\,z'$, thus resulting in the following computation that cannot be completed in the forward direction:

$$\begin{array}{rl}
& m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o\,.\,(z_{\,q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o\,.\,z\,.\,o'.\,o' + o\,.\,o\,.\,o'.\,z') \\
\xrightarrow{o[3]}_{\mathtt{a}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o[3]\,.\,(z_{\,q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o\,.\,z\,.\,o'.\,o' + o[3]\,.\,o\,.\,o'.\,z') \\
\xrightarrow{(q_2)^{[4]}}_{\mathtt{p}} & m[1]\,.\,(z\,.\,(z_{\,q_1} \oplus o)_{\,p \oplus_{[2]}} o[3]\,.\,(z_{\,[4]q_2} \oplus o)) \parallel_L \\
& m[1]\,.\,(z\,.\,z\,.\,z'.\,z' + z\,.\,o\,.\,z'.\,o' + o\,.\,z\,.\,o'.\,o' + o[3]\,.\,o\,.\,o'.\,z')
\end{array}$$

This can be undone – in favor of the third branch of the aforementioned nondeterministic choice, i.e., $o \,.\, z \,.\, o' \,.\, z'$ – thanks to the fact RPPC is reversible:

$$\begin{array}{rl} & m[1] \,.\, (z \,.\, (z \,{}_{q_1}\!\oplus o) \,{}_{p}\!\oplus_{[2]} o[3] \,.\, (z \,{}_{[4]q_2}\!\oplus o)) \,\|_L \\ & m[1] \,.\, (z \,.\, z \,.\, z' \,.\, z' + z \,.\, o \,.\, z' \,.\, o' + o \,.\, z \,.\, o' \,.\, o' + o[3] \,.\, o \,.\, o' \,.\, z') \\ \xrightarrow{(q_2)^{[4]}}_{\mathtt{p}} & m[1] \,.\, (z \,.\, (z \,{}_{q_1}\!\oplus o) \,{}_{p}\!\oplus_{[2]} o[3] \,.\, (z \,{}_{q_2}\!\oplus o)) \,\|_L \\ & m[1] \,.\, (z \,.\, z \,.\, z' \,.\, z' + z \,.\, o \,.\, z' \,.\, o' + o \,.\, z \,.\, o' \,.\, o' + o[3] \,.\, o \,.\, o' \,.\, z') \\ \xrightarrow{o[3]}_{\mathtt{a}} & m[1] \,.\, (z \,.\, (z \,{}_{q_1}\!\oplus o) \,{}_{p}\!\oplus_{[2]} o \,.\, (z \,{}_{q_2}\!\oplus o)) \,\|_L \\ & m[1] \,.\, (z \,.\, z \,.\, z' \,.\, z' + z \,.\, o \,.\, z' \,.\, o' + o \,.\, z \,.\, o' \,.\, o' + o \,.\, o \,.\, o' \,.\, z') \end{array}$$

Since our calculus is fully reversible, i.e., every action can be undone in RPPC, even m can be reverted although measurement is known to be an irreversible operation in quantum computing. To amend this, following [11] we should see m as an irreversible action, to which only forward semantic rules are applicable.

4 Causal Reversibility of RPPC

We now prove the causal reversibility of RPPC. This means that each reachable process of RPPC is able to backtrack *correctly*, i.e., without encountering previously inaccessible states, and *flexibly*, i.e., along any path that is causally equivalent to the one undertaken in the forward direction. This is accomplished through the notion of concurrent transitions of [10] and the technique of [29].

A necessary condition for reversibility is the *loop property* [10,36,29]. It establishes that each executed action can be undone and that each undone action can be redone, which in our setting needs to be extended to probabilistic selections. Therefore, when considering the states associated with two reachable processes, either there is no transition between them, or there is a pair of identically labeled transitions such that one is a forward transition from the first to the second state while the other is a backward transition from the second to the first state.

Proposition 1 (loop property). *Let $R, S \in \mathbb{P}$ and $\ell \in \mathcal{L} = \mathcal{A}_{\mathcal{K}} \cup \mathbb{R}_{\mathcal{K}}$. Then $R \xrightarrow{\ell} S$ iff $S \xrightarrow{\ell} R$.* ∎

Given a transition $\theta : R \overset{\ell}{\longmapsto} S$ with $R, S \in \mathbb{P}$, we call R the *source* of θ and S its *target*. If θ is a forward transition, i.e., $\theta : R \xrightarrow{\ell} S$, we denote by $\overline{\theta} : S \xrightarrow{\ell} R$ the corresponding backward transition. Two transitions are said to be *coinitial* if they have the same source and *cofinal* if they have the same target. Two transitions are *composable* when the target of the first transition coincides with the source of the second transition. A finite sequence of pairwise composable transitions is called a *path*. We use ε for the empty path and ω to range over paths, with $|\omega|$ denoting the length of ω, i.e., the number of transitions constituting ω. When ω is a forward path, we denote by $\overline{\omega}$ the corresponding backward path, where the order of the transitions is reversed. The notions of source, target, coinitiality, cofinality, and composability naturally extend to paths. We indicate with $\omega_1\omega_2$ the composition of the two paths ω_1 and ω_2 when they are composable.

Before specifying when two transitions are concurrent [10], we need to present the notion of process context along with the set of causes – identified by action and probabilistic keys – leading to a given communication key.

A *process context* is a process with a hole $\bullet$ in it, generated by the grammar:

$$\mathcal{C} ::= \bullet \mid a[i]\,.\,\mathcal{C} \mid \mathcal{C} \,{}_{[i]p}\!\oplus R \mid R \,{}_{p}\!\oplus_{[i]} \mathcal{C} \mid \mathcal{C} + R \mid R + \mathcal{C} \mid \mathcal{C} \parallel_L R \mid R \parallel_L \mathcal{C}$$

We write $\mathcal{C}[S]$ to denote the process obtained by replacing the hole in $\mathcal{C}$ with S.

The *causal set* $\mathtt{cau}(R,i)$ of $R \in \mathbb{P}$ until $i \in \mathcal{K}$ under $\mathtt{key_a}(R) \cap \mathtt{key_p}(R) = \emptyset$ is inductively defined as:

$$\begin{array}{rcl}
\mathtt{cau}(F,i) & = & \emptyset \\
\mathtt{cau}(a[j]\,.\,R,i) & = & \begin{cases} \emptyset & \text{if } j = i \text{ or } i \notin \mathtt{key_a}(R) \cup \mathtt{key_p}(R) \\ \{j\} \cup \mathtt{cau}(R,i) & \text{otherwise} \end{cases} \\
\mathtt{cau}(R \,{}_{[j]p}\!\oplus S,i) & = & \begin{cases} \emptyset & \text{if } j = i \text{ or } i \notin \mathtt{key_a}(R) \cup \mathtt{key_p}(R) \\ \{j\} \cup \mathtt{cau}(R,i) & \text{otherwise} \end{cases} \\
\mathtt{cau}(R \,{}_{p}\!\oplus_{[j]} S,i) & = & \begin{cases} \emptyset & \text{if } j = i \text{ or } i \notin \mathtt{key_a}(S) \cup \mathtt{key_p}(S) \\ \{j\} \cup \mathtt{cau}(S,i) & \text{otherwise} \end{cases} \\
\mathtt{cau}(R + S,i) & = & \mathtt{cau}(R,i) \cup \mathtt{cau}(S,i) \\
\mathtt{cau}(R \parallel_L S,i) & = & \mathtt{cau}(R,i) \cup \mathtt{cau}(S,i)
\end{array}$$

If $i \in \mathtt{key_a}(R) \cup \mathtt{key_p}(R)$, then $\mathtt{cau}(R,i)$ represents the set of keys in R that caused i, with $\mathtt{cau}(R,i) \subset \mathtt{key_a}(R) \cup \mathtt{key_p}(R)$ as on the one hand $i \notin \mathtt{cau}(R,i)$ and on the other hand keys that are not causally related to i are not considered. Key j causes key i if it appears before i in R, i.e., if i is inside the scope of j.

We are now in a position to define, for coinitial transitions, what we mean by concurrent transitions on the basis of the notion of conflicting transitions. As in previous work, the first condition below tells that a forward transition is in conflict with a coinitial backward one whenever the latter tries to undo a cause of the key of the former (with abuse of notation the key is made explicit next to ℓ), while the second one deems as conflictual two action transitions respectively generated by the two subprocesses of a nondeterministic choice. The further third condition is an adaptation of the previous one to the probabilistic case. Since probabilistic choices have to be resolved before nondeterministic ones, there can never be conflicts between action transitions and probabilistic transitions.

Definition 1 (conflicting and concurrent transitions). *Two coinitial transitions θ_1 and θ_2 from a process $R \in \mathbb{P}$ are* in conflict *if one of the following conditions holds, otherwise they are said to be* concurrent*:*

- $\theta_1 : R \xrightarrow{\ell_1[i]} S_1$ *and* $\theta_2 : R \dashrightarrow S_2$ *with* $j \in \mathtt{cau}(S_1,i)$.
- $R = \mathcal{C}[F_1 + F_2]$ *with* θ_k *deriving in* R *from* $F_k \xrightarrow{a_k[i_k]}_{\mathtt{a}} S_k$ *for* $k = 1,2$.
- $R = \mathcal{C}[F_1 \,{}_{p}\!\oplus F_2]$ *with* θ_k *deriving in* R *from* $F_k \xrightarrow{(p_k)^{[i_k]}}_{\mathtt{p}} S_k$ *for* $k = 1,2$. ■

We prove causal reversibility by adapting the technique of [29], according to which causal consistency stems from the *square property* – which amounts to concurrent transitions being confluent – *backward transitions independence* – which generalizes the concept of backward determinism used for reversible sequential

languages [49] – and *past well foundedness* – which ensures that reachable processes have a finite past.

Before proving the three properties, as for the classical definition of square property [29] we have to deal with the fact that probabilistic choices take precedence over nondeterministic ones in the forward direction. Hence, even if from a process there are two transitions coming from the two subprocesses of a parallel composition, we have to determine whether either reached process can directly perform the other transition to close the square, or it has to first resolve probabilistic choices. Once such choices have been resolved, the remaining transition can be done thus closing the square. For example, consider $a \,.\, (c \,.\, \underline{0}\,{}_p\!\oplus\, \underline{0}) \parallel_\emptyset b \,.\, \underline{0}$ depicted in Figure 2. It can initially perform an a-action and a b-action, but if a is done then a probabilistic choice is enabled. Hence, before doing b, the process has to first resolve the probabilistic choice and only then it can proceed with b. On the other hand, if the process does b first, then it can immediately do a, but in order to reach the same process as the left path it also has to resolve the probabilistic choice. As an analogous though more extended example of square, consider the one originated from $a_1 \,.\, (b \,.\, \underline{0}\,{}_p\!\oplus\, c \,.\, \underline{0}) \parallel_\emptyset a_2 \,.\, (d \,.\, \underline{0}\,{}_q\!\oplus\, e \,.\, \underline{0})$. These cases are handled by the second and the third clauses below.

Lemma 1 (square property). *Let $\theta_1 : R \stackrel{\ell_1}{\longmapsto} S_1$ and $\theta_2 : R \stackrel{\ell_2}{\longmapsto} S_2$ be two coinitial transitions from a process $R \in \mathbb{P}$. If θ_1 and θ_2 are concurrent, then one of the following holds:*

- *If $S_1 \not\rightarrow_{\mathtt{p}}$ and $S_2 \not\rightarrow_{\mathtt{p}}$ then there exist two cofinal transitions $\theta_2' : S_1 \stackrel{\ell_2}{\longmapsto} S$ and $\theta_1' : S_2 \stackrel{\ell_1}{\longmapsto} S$ with $S \in \mathbb{P}$.*
- *If $S_1 \not\rightarrow_{\mathtt{p}}$ and $S_2 \rightarrow_{\mathtt{p}}$ then there exist two cofinal paths $\omega_2' : S_1 \stackrel{\ell_2}{\longmapsto} S_p \stackrel{\ell_p}{\longmapsto} S$ and $\omega_1' : S_2 \stackrel{\ell_p}{\longmapsto} S_p \stackrel{\ell_1}{\longmapsto} S$ with $S_p, S \in \mathbb{P}$.*
- *If $S_1 \rightarrow_{\mathtt{p}}$ and $S_2 \rightarrow_{\mathtt{p}}$ then there exist two cofinal paths $\omega_2' : S_1 \stackrel{\ell_p}{\longmapsto} S_p^1 \stackrel{\ell_2}{\longmapsto} S_q^1 \stackrel{\ell_q}{\longmapsto} S$ and $\omega_1' : S_2 \stackrel{\ell_q}{\longmapsto} S_q^2 \stackrel{\ell_1}{\longmapsto} S_p^2 \stackrel{\ell_p}{\longmapsto} S$ with $S_p^1, S_p^2, S_q^1, S_q^2, S \in \mathbb{P}$.* ■

Lemma 2 (backward transitions independence). *Let $R \in \mathbb{P}$. Then two coinitial backward transitions $\theta_1 : R \stackrel{\ell_1}{\dashrightarrow} S_1$ and $\theta_2 : R \stackrel{\ell_2}{\dashrightarrow} S_2$ are concurrent.* ■

Lemma 3 (past well foundedness). *Let $R_0 \in \mathbb{P}$. Then there is no infinite sequence of backward transitions such that $R_i \stackrel{\ell_i}{\dashrightarrow} R_{i+1}$ for all $i \in \mathbb{N}$.* ■

Following [10,32], we also define a notion of *causal equivalence* over paths. In addition to identifying the composition of a transition and its inverse with the empty path, it abstracts from the order of concurrent action and probabilistic transitions. In this way, paths obtained by swapping the order of those transitions are identified with each other.

Since probabilistic choices take precedence over nondeterministic ones in the forward direction, a swap between two concurrenct action transitions is not always possible, unless all probabilistic choices have been made. More precisely,

after opening a square, it may be the case that, in order to close it, the process has to first resolve some probabilistic choices. This is rendered by the third and fourth clauses in the definition below. For example, if we consider again process $a\,.\,(c\,.\,\underline{0}\,{}_p\!\oplus\,\underline{0})\,\|_\emptyset\,b\,.\,\underline{0}$ in Figure 2, we have that the action transitions a and b are concurrent and coinitial. If we take the left path, then after doing a the process has to resolve the probabilistic choice. Suppose it decides for the right branch, which is labeled with $(1-p)$. Then a process is reached in which b can be done. On the other hand, if we take the right path, we have that the process can do b followed by a but then again, in order to close the square, it has to resolve the probabilistic choice; if it decides for $(1-p)$ the same process as the left path is reached. Therefore the two paths can be considered as causally equivalent. Something similar happens along the more extended square originating from $a_1\,.\,(b\,.\,\underline{0}\,{}_p\!\oplus\,c\,.\,\underline{0})\,\|_\emptyset\,a_2\,.\,(d\,.\,\underline{0}\,{}_q\!\oplus\,e\,.\,\underline{0})$.

Definition 2 (causal equivalence). Causal equivalence $\asymp$ *is the smallest equivalence relation over paths that is closed under composition and satisfies the following clauses:*

1. $\theta_1\theta'_2 \asymp \theta_2\theta'_1$ *for every two coinitial concurrent action transitions* $\theta_1 : R \stackrel{\ell_1}{\longmapsto} R_1$ *and* $\theta_2 : R \stackrel{\ell_2}{\longmapsto} R_2$ *and every two cofinal action transitions* $\theta'_2 : R_1 \stackrel{\ell_2}{\longmapsto} S$ *and* $\theta'_1 : R_2 \stackrel{\ell_1}{\longmapsto} S$ *respectively composable with the previous ones.*
2. $\theta\overline{\theta} \asymp \varepsilon$ *and* $\overline{\theta}\theta \asymp \varepsilon$ *for every transition* θ*.*
3. $\theta_1\theta_p\theta'_2 \asymp \theta_2\theta'_1\theta'_p$ *for every two coinitial concurrent action transitions* $\theta_1 : R \stackrel{\ell_1}{\longmapsto} R_1$ *and* $\theta_2 : R \stackrel{\ell_2}{\longmapsto} R_2$*, every probabilistic transition* $\theta_p : R_1 \stackrel{\ell_p}{\longmapsto} R'_1$*, and every two cofinal transitions* $\theta'_2 : R'_1 \stackrel{\ell_2}{\longmapsto} S$ *and* $\theta'_p : R'_2 \stackrel{\ell_p}{\longmapsto} S$*, with* $\theta'_1 : R_2 \stackrel{\ell_p}{\longmapsto} R'_2$*.*
4. $\theta_1\theta_p\theta'_2\theta'_q \asymp \theta_2\theta_q\theta'_1\theta'_p$ *for every two coinitial concurrent action transitions* $\theta_1 : R \stackrel{\ell_1}{\longmapsto} R_1$ *and* $\theta_2 : R \stackrel{\ell_2}{\longmapsto} R_2$*, every two probabilistic transitions* $\theta_p : R_1 \stackrel{\ell_p}{\longmapsto} R'_1$ *and* $\theta_q : R_2 \stackrel{\ell_q}{\longmapsto} R'_2$*, and every two cofinal transitions* $\theta'_q : R'_2 \stackrel{\ell_q}{\longmapsto} S$ *and* $\theta'_p : R'_1 \stackrel{\ell_p}{\longmapsto} S$*, with* $\theta'_1 : R_2 \stackrel{\ell_1}{\longmapsto} R'_2$ *and* $\theta'_2 : R_1 \stackrel{\ell_2}{\longmapsto} R'_1$*.* ■

The further property below, called the *parabolic lemma* in [29], states that every path can be seen as a backward path followed by a forward path. As observed in [10], up to causal equivalence one can always reach for the maximum freedom of choice among transitions by going backward and only then going forward (not the other way around). Intuitively, computations can be viewed as parabolas: the system first draws potential energy from its memory by undoing all the executed actions and then restarts. The proof of the parabolic lemma has to account for the presence of probabilistic transitions.

Lemma 4 (parabolic lemma). *For each path* ω*, there exist two forward paths* ω_1 *and* ω_2 *such that* $\omega \asymp \overline{\omega_1}\omega_2$ *and* $|\omega_1| + |\omega_2| \leq |\omega|$*.* ■

We conclude by obtaining a property called *causal consistency* in [29], which establishes that being coinitial and cofinal is necessary and sufficient in order for two paths to be causally equivalent, i.e., to contain concurrent action and probabilistic transitions in different orders (swap) or to be one the empty path and the other a transition followed by its reverse (cancelation).

Theorem 1 (causal consistency). *Let ω_1 and ω_2 be two paths. Then $\omega_1 \asymp \omega_2$ iff ω_1 and ω_2 are both coinitial and cofinal.* ■

Theorem 1 shows that causal equivalence characterizes a space for admissible rollbacks that are (i) correct as they do not lead to states not reachable by any forward path and (ii) flexible enough to allow undo operations to be rearranged with respect to the order in which the undone concurrent actions and probabilistic transitions were originally performed. This implies that the states reached by any backward path could be reached by performing forward paths only. Thus, we can conclude that RPPC meets causal reversibility.

5 Conclusions

In this paper we have studied causal reversibility [10] of nondeterministic and probabilistic process calculi in the non-strictly alternating model [20,35]. The syntax and operational semantics have been defined by suitably adapting the method of [36], while causal reversibility has been demonstrated by suitably adapting the technique of [29], thus extending results developed in the fully nondeterministic setting.

As future work, similar to the stochastically timed case [5,4], for our reversible probabilistic calculus we plan to study behavioral equivalences as well as time reversibility [22] and its possible relationships with causal reversibility.

Acknowledgments. This work has been supported by the Italian MUR PRIN 2020 project *NiRvAna*, the Italian MUR PRIN 2022 project *DeKLA*, the French ANR project *DCore*, and the Italian INdAM-GNCS project *RISICO*.

References

1. Andova, S., Georgievska, S., Trcka, N.: Branching bisimulation congruence for probabilistic systems. Theoretical Computer Science **413**, 58–72 (2012)
2. Bennett, C.H.: Logical reversibility of computation. IBM Journal of Research and Development **17**, 525–532 (1973)
3. Bernardo, M., Esposito, A.: Modal logic characterizations of forward, reverse, and forward-reverse bisimilarities. In: Proc. of the 14th Int. Symp. on Games, Automata, Logics, and Formal Verification (GANDALF 2023). EPTCS, vol. 390, pp. 67–81 (2023)
4. Bernardo, M., Lanese, I., Marin, A., Mezzina, C.A., Rossi, S., Sacerdoti Coen, C.: Causal reversibility implies time reversibility. In: Proc. of the 20th Int. Conf. on the Quantitative Evaluation of Systems (QEST 2023). LNCS, vol. 14287, pp. 270–287. Springer (2023)

5. Bernardo, M., Mezzina, C.A.: Bridging causal reversibility and time reversibility: A stochastic process algebraic approach. Logical Methods in Computer Science **19(2)**, 6:1–6:27 (2023)
6. Bernardo, M., Mezzina, C.A.: Causal reversibility for timed process calculi with lazy/eager durationless actions and time additivity. In: Proc. of the 21st Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS 2023). LNCS, vol. 14138, pp. 15–32. Springer (2023)
7. Bérut, A., Arakelyan, A., Petrosyan, A., Ciliberto, S., Dillenschneider, R., Lutz, E.: Experimental verification of Landauer's principle linking information and thermodynamics. Nature **483**, 187–189 (2012)
8. Cattani, S., Segala, R.: Decision algorithms for probabilistic bisimulation. In: Proc. of the 13th Int. Conf. on Concurrency Theory (CONCUR 2002). LNCS, vol. 2421, pp. 371–385. Springer (2002)
9. Chatterjee, K., Goharshady, A.K., Pourdamghani, A.: Probabilistic smart contracts: Secure randomness on the blockchain. In: Proc. of the 1st IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC 2019). pp. 403–412. IEEE-CS Press (2019)
10. Danos, V., Krivine, J.: Reversible communicating systems. In: Proc. of the 15th Int. Conf. on Concurrency Theory (CONCUR 2004). LNCS, vol. 3170, pp. 292–307. Springer (2004)
11. Danos, V., Krivine, J.: Transactions in RCCS. In: Proc. of the 16th Int. Conf. on Concurrency Theory (CONCUR 2005). LNCS, vol. 3653, pp. 398–412. Springer (2005)
12. De Nicola, R., Montanari, U., Vaandrager, F.: Back and forth bisimulations. In: Proc. of the 1st Int. Conf. on Concurrency Theory (CONCUR 1990). LNCS, vol. 458, pp. 152–165. Springer (1990)
13. Derman, C.: Finite State Markovian Decision Processes. Academic Press (1970)
14. Esposito, A., Aldini, A., Bernardo, M.: Branching bisimulation semantics enables noninterference analysis of reversible systems. In: Proc. of the 43rd Int. Conf. on Formal Techniques for Distributed Objects, Components, and Systems (FORTE 2023). LNCS, vol. 13910, pp. 57–74. Springer (2023)
15. Esposito, A., Aldini, A., Bernardo, M.: Noninterference analysis of reversible probabilistic systems. In: Proc. of the 44th Int. Conf. on Formal Techniques for Distributed Objects, Components, and Systems (FORTE 2024). LNCS, vol. 14678, pp. 39–59. Springer (2024)
16. Frank, M.P.: Physical foundations of Landauer's principle. In: Proc. of the 10th Int. Conf. on Reversible Computation (RC 2018). LNCS, vol. 11106, pp. 3–33. Springer (2018)
17. Giachino, E., Lanese, I., Mezzina, C.A.: Causal-consistent reversible debugging. In: Proc. of the 17th Int. Conf. on Fundamental Approaches to Software Engineering (FASE 2014). LNCS, vol. 8411, pp. 370–384. Springer (2014)
18. van Glabbeek, R.J., Smolka, S.A., Steffen, B.: Reactive, generative and stratified models of probabilistic processes. Information and Computation **121**, 59–80 (1995)
19. Hansson, H.: Time and Probability in Formal Design of Distributed Systems. PhD Thesis (1992)
20. Hansson, H., Jonsson, B.: A calculus for communicating systems with time and probabilities. In: Proc. of the 11th IEEE Real-Time Systems Symp. (RTSS 1990). pp. 278–287. IEEE-CS Press (1990)
21. Hoare, C.A.R.: Communicating Sequential Processes. Prentice Hall (1985)
22. Kelly, F.P.: Reversibility and Stochastic Networks. John Wiley & Sons (1979)

23. Kemeny, J.G., Snell, J.L.: Finite Markov Chains. Van Nostrand (1960)
24. Krivine, J.: A verification technique for reversible process algebra. In: Proc. of the 4th Int. Workshop on Reversible Computation (RC 2012). LNCS, vol. 7581, pp. 204–217. Springer (2012)
25. Landauer, R.: Irreversibility and heat generation in the computing process. IBM Journal of Research and Development **5**, 183–191 (1961)
26. Lanese, I., Lienhardt, M., Mezzina, C.A., Schmitt, A., Stefani, J.B.: Concurrent flexible reversibility. In: Proc. of the 22nd European Symp. on Programming (ESOP 2013). LNCS, vol. 7792, pp. 370–390. Springer (2013)
27. Lanese, I., Medić, D., Mezzina, C.A.: Static versus dynamic reversibility in CCS. Acta Informatica **58**, 1–34 (2021)
28. Lanese, I., Nishida, N., Palacios, A., Vidal, G.: CauDEr: A causal-consistent reversible debugger for Erlang. In: Proc. of the 14th Int. Symp. on Functional and Logic Programming (FLOPS 2018). LNCS, vol. 10818, pp. 247–263. Springer (2018)
29. Lanese, I., Phillips, I., Ulidowski, I.: An axiomatic theory for reversible computation. ACM Trans. on Computational Logic **25(2)**, 11:1–11:40 (2024)
30. Laursen, J.S., Ellekilde, L.P., Schultz, U.P.: Modelling reversible execution of robotic assembly. Robotica **36**, 625–654 (2018)
31. Lehmann, D., Rabin, M.O.: On the advantage of free choice: A symmetric and fully distributed solution to the dining philosophers problem. In: Proc. of the 8th ACM Symp. on Principles of Programming Languages (POPL 1981). pp. 133–138. ACM Press (1981)
32. Lévy, J.J.: An algebraic interpretation of the $\lambda\beta$K-calculus; and an application of a labelled λ-calculus. Theoretical Computer Science **2**, 97–114 (1976)
33. Milner, R.: Communication and Concurrency. Prentice Hall (1989)
34. Perumalla, K.S., Park, A.J.: Reverse computation for rollback-based fault tolerance in large parallel systems - Evaluating the potential gains and systems effects. Cluster Computing **17**, 303–313 (2014)
35. Philippou, A., Lee, I., Sokolsky, O.: Weak bisimulation for probabilistic systems. In: Proc. of the 11th Int. Conf. on Concurrency Theory (CONCUR 2000). LNCS, vol. 1877, pp. 334–349. Springer (2000)
36. Phillips, I., Ulidowski, I.: Reversing algebraic process calculi. Journal of Logic and Algebraic Programming **73**, 70–96 (2007)
37. Phillips, I., Ulidowski, I., Yuen, S.: A reversible process calculus and the modelling of the ERK signalling pathway. In: Proc. of the 4th Int. Workshop on Reversible Computation (RC 2012). LNCS, vol. 7581, pp. 218–232. Springer (2012)
38. Pinna, G.M.: Reversing steps in membrane systems computations. In: Proc. of the 18th Int. Conf. on Membrane Computing (CMC 2017). LNCS, vol. 10725, pp. 245–261. Springer (2017)
39. Prabhu, P., Ramalingam, G., Vaswani, K.: Safe programmable speculative parallelism. In: Proc. of the 31st ACM Conf. on Programming Language Design and Implementation (PLDI 2010). pp. 50–61. ACM Press (2010)
40. Rabin, M.O.: Probabilistic automata. Information and Control **6**, 230–245 (1963)
41. Schordan, M., Oppelstrup, T., Jefferson, D.R., Barnes Jr., P.D.: Generation of reversible C++ code for optimistic parallel discrete event simulation. New Generation Computing **36**, 257–280 (2018)
42. Segala, R.: Modeling and Verification of Randomized Distributed Real-Time Systems. PhD Thesis (1995)

43. Segala, R., Lynch, N.A.: Probabilistic simulations for probabilistic processes. In: Proc. of the 5th Int. Conf. on Concurrency Theory (CONCUR 1994). LNCS, vol. 836, pp. 481–496. Springer (1994)
44. Segala, R., Turrini, A.: Comparative analysis of bisimulation relations on alternating and non-alternating probabilistic models. In: Proc. of the 2nd Int. Conf. on the Quantitative Evaluation of Systems (QEST 2005). pp. 44–53. IEEE-CS Press (2005)
45. Siljak, H., Psara, K., Philippou, A.: Distributed antenna selection for massive MIMO using reversing Petri nets. IEEE Wireless Communication Letters **8**, 1427–1430 (2019)
46. Turrini, A., Hermanns, H.: Polynomial time decision algorithms for probabilistic automata. Information and Computation **244**, 134–171 (2015)
47. Vassor, M., Stefani, J.B.: Checkpoint/rollback vs causally-consistent reversibility. In: Proc. of the 10th Int. Conf. on Reversible Computation (RC 2018). LNCS, vol. 11106, pp. 286–303. Springer (2018)
48. de Vries, E., Koutavas, V., Hennessy, M.: Communicating transactions. In: Proc. of the 21st Int. Conf. on Concurrency Theory (CONCUR 2010). LNCS, vol. 6269, pp. 569–583. Springer (2010)
49. Yokoyama, T., Glück, R.: A reversible programming language and its invertible self-interpreter. In: Proc. of the 13th ACM Workshop on Partial Evaluation and Semantics-based Program Manipulation (PEPM 2007). pp. 144–153. ACM Press (2007)

A Proofs of Results

Proof of Proposition 1.
We proceed by induction on the depth of the derivation of either transition and then by case analysis on the last applied rule. Since each forward (resp. backward) rule in Tables 2 and 3 has a fully corresponding backward (resp. forward) rule in the same table except for PPAR and PPAR$^\bullet$, we concentrate on the last mentioned pair of rules.

Consider $R \parallel_L S$ and let $R \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R'$ with $i \notin \mathtt{key}_{\mathtt{p}}(S)$ and $S \not\rightarrow_{\mathtt{p}}$, so that the application of PPAR yields $R \parallel_L S \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R' \parallel_L S$. By applying the induction hypothesis to $R \xrightarrow{(p)^{[i]}}_{\mathtt{p}} R'$ we obtain $R' \xdashrightarrow{(p)^{[i]}}_{\mathtt{p}} R$. In order to apply PPAR$^\bullet$, we also need to have $\mathtt{npa}(S) \vee \neg\mathtt{npa}(R')$, which means that S has no past actions or R' has them. Since we are considering reachable processes and the semantics stipulates that in the forward direction all probabilistic choices have to resolved at once and before nondeterministic choices, the probabilistic choice in R is (i) in the scope of an executed action prefix or (ii) just a top level probabilistic choice; moreover, S enables (iii) at least one action or (iv) no actions at all. There are four cases depending on the combination of the forms of R and S. We will consider only the one in which the probabilistic choice of R is in the scope of an executed action prefix and S has an enabled action, which is the case combining (i) and (iii); the other cases are similar. Thus we have $R = \mathcal{C}_1[a[k] \,.\, (R_1 \,{}_p\!\oplus\, R_2)]$ and $S = \mathcal{C}_2[b \,.\, S']$, so by applying PPAR we obtain $\mathcal{C}_1[a[k] \,.\, (R_1 \,{}_p\!\oplus\, R_2)] \parallel_L \mathcal{C}_2[b \,.\, S'] \xrightarrow{(p)^{[i]}}_{\mathtt{p}} \mathcal{C}_1[a[k] \,.\, (R_1 \,{}_{[i]p}\!\oplus\, R_2)] \parallel_L \mathcal{C}_2[b \,.\, S']$. Clearly $\neg\mathtt{npa}(R')$ holds, so that PPAR$^\bullet$ can be applied and we obtain $R' \parallel_L S \xdashrightarrow{(p)^{[i]}}_{\mathtt{p}} R \parallel_L S$ as desired.
The case in which we first apply PPAR$^\bullet$ and then PPAR is simpler because the conditions of PPAR are less demanding than those of PPAR$^\bullet$. ■

Proof of Lemma 1.
We distinguish among three cases based on the direction of θ_1 and θ_2:

- If θ_1 and θ_2 are both forward, there are three subcases:
  - If their labels are actions, since θ_1 and θ_2 are concurrent they cannot originate from a choice operator by virtue of condition 2 of Definition 1. They must thus be generated by a parallel composition, but not through rule COO because θ_1 and θ_2 must have different keys and hence cannot synchronize. Without loss of generality, we can assume that $R = R_1 \parallel_L R_2$ with $R_1 \xrightarrow{a[i]}_{\mathtt{a}} S_1$, $R_2 \xrightarrow{b[j]}_{\mathtt{a}} S_2$, $a, b \notin L$, and $i \neq j$. There are three further subcases:
    - If neither S_1 nor S_2 enables a probabilistic choice, by applying rule PAR we obtain $R_1 \parallel_L R_2 \xrightarrow{a[i]}_{\mathtt{a}} S_1 \parallel_L R_2 \xrightarrow{b[j]}_{\mathtt{a}} S_1 \parallel_L S_2$ as well as $R_1 \parallel_L R_2 \xrightarrow{b[j]}_{\mathtt{a}} R_1 \parallel S_2 \xrightarrow{a[i]}_{\mathtt{a}} S_1 \parallel_L S_2$. This satisfies the first clause of the square property.

  - If S_2 enables a probabilistic choice whereas S_1 does not, that choice has to be resolved before letting R_1 do its action transition. Suppose that $S_2 \xrightarrow{(p)^{[k]}}_{\mathtt{p}} S_2'$ with $k \notin \mathtt{key_a}(S_2)$. By applying rule PPar we have that $R_1 \parallel_L S_2 \xrightarrow{(p)^{[k]}}_{\mathtt{p}} R_1 \parallel_L S_2'$. Now since there are no more enabled probabilistic choices, as they have to be resolved at once even if they are nested in S_1, by applying rule Par we have that $R_1 \parallel_L S_2' \xrightarrow{b[j]}_{\mathtt{a}} S_1 \parallel_L S_2'$. This satisfies the second clause of the square property.
  - If both S_1 and S_2 enable a probabilistic choice, then we proceed like in the previous subcase. This satisfies the third clause of the square property.
- If their labels are probabilities, this case never applies because it is impossible to derive two concurrent probabilistic transitions from the same process.
- If one label is an action and the other is a probability, this case cannot happen because the probabilistic choices have to be resolved first, hence the action transition cannot be generated.

– If θ_1 and θ_2 are both backward, there are three subcases:

- If their labels are actions, then we proceed like in the first subcase of the case in which θ_1 and θ_2 are both forward.
- If their labels are probabilities, say $S_1 \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{a}} R_1$ and $S_2 \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{a}} R_2$, since $i \neq j$ the only rule that can be applied is PPar$^\bullet$. This implies that $(\mathtt{npa}(S_2) \vee \neg\mathtt{npa}(S_1)) \wedge S_2 \not\rightarrow_{\mathtt{p}}$ holds for the first transition and $(\mathtt{npa}(S_1) \vee \neg\mathtt{npa}(S_2)) \wedge S_1 \not\rightarrow_{\mathtt{p}}$ holds for the second one. The only case in which both transitions can be generated from the same process is when $S_i \not\rightarrow_{\mathtt{p}}$ and $\neg\mathtt{npa}(S_i)$ for $i \in \{1,2\}$; the other cases are ruled out by the semantics. By applying rule PPar$^\bullet$ we have that $S_1 \parallel_L S_2 \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{p}} R_1 \parallel_L S_2$ and $S_1 \parallel_L S_2 \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} S_1 \parallel_L R_2$. Now we have that R_1 cannot be a standard process and has at least one backward action transition to do, otherwise $\mathtt{npa}(S_1)$ would hold, but this violates the hypothesis. Hence from $R_1 \overset{a[k]}{\dashrightarrow}_{\mathtt{a}} R_1'$ we can derive $R_1 \parallel_L S_2 \overset{a[k]}{\dashrightarrow}_{\mathtt{a}} R_1' \parallel_L S_2$ by applying rule Par$^\bullet$. Also, note that since the undo of an action never enables a probabilistic choice we still have that $\mathtt{npa}(R_1')$. We can apply the same reasoning to S_2 and from $S_2 \overset{q^{[j]}}{\dashrightarrow}_{\mathtt{p}} R_2 \overset{b[h]}{\dashrightarrow}_{\mathtt{a}} R_2'$ by appyling PPar$^\bullet$ and Par$^\bullet$ we derive $R_1' \parallel_L S_2 \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} R_1' \parallel_L R_2 \overset{b[h]}{\dashrightarrow}_{\mathtt{a}} R_1' \parallel_L R_2'$. If we start the execution from R_2, with the same reasoning as above we obtain the following execution: $S_1 \parallel_L S_2 \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} \overset{b[h]}{\dashrightarrow}_{\mathtt{a}} S_1 \parallel_L R_2' \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{p}} \overset{a[k]}{\dashrightarrow}_{\mathtt{a}} R_1' \parallel_L R_2'$. This satisfies the third clause of the square property.

- If one label is an action and the other is a probability, say $S_1 \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{p}} R_1$ and $S_2 \overset{a[j]}{\dashrightarrow}_{\mathtt{a}} R_2$, then $S_2 \not\dashrightarrow_{\mathtt{p}}$ otherwise the backward transition of the left subprocess would not be possible. By using first PPAR$^\bullet$ and then PAR$^\bullet$ we obtain $S_1 \parallel_L S_2 \overset{(q)^{[i]}}{\dashrightarrow}_{\mathtt{p}} R_1 \parallel_L S_2 \overset{a[j]}{\dashrightarrow}_{\mathtt{a}} R_1 \parallel_L R_2$. On the other hand, by using first PAR$^\bullet$ and then PPAR$^\bullet$ we obtain $S_1 \parallel_L S_2 \overset{a[j]}{\dashrightarrow}_{\mathtt{a}} S_1 \parallel_L R_2 \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{p}} R_1 \parallel_L R_2$. This satisfies the second clause of the square property.

– If θ_1 is forward and θ_2 is backward, there are three subcases:
  - If their labels are actions, then we proceed like in the first subcase of the case in which θ_1 and θ_2 are both forward.
  - If their labels are probabilities, say $S_1 \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{p}} R_1$ and $S_2 \overset{(q)^{[j]}}{\longrightarrow}_{\mathtt{p}} R_2$, since the two transitions use different keys the only rule that can be applied is PPAR$^\bullet$. This implies that $(\mathtt{npa}(S_2) \vee \neg\mathtt{npa}(S_1)) \wedge S_2 \not\dashrightarrow_{\mathtt{p}}$. Since S_2 has a forward probabilistic transition, the condition does not hold and hence this case is ruled out.
  - If one label is an action and the other is a probability, the case cannot happen because from this state the undoing of a probabilistic choice is forbidden. ■

Proof of Lemma 2
By Definition 1 it is not possible for two backward transitions to be in conflict. ■

Proof of Lemma 3.
By induction on $|\mathtt{key}_{\mathtt{a}}(R_0)|$. Indeed, every backward transition between two different processes decreases by one the total number of past actions, with this number being finite. ■

Proof of Lemma 4
Let $d(\omega)$ be the number of discording pairs within path ω, where two forward transitions θ_1 and θ_2 form a discording pair iff $\theta_1\theta_1^p$ and $\overline{\theta_2}\overline{\theta_2^q}$ occur next to each other in that order inside ω, with θ_1^p and θ_2^q possibly empty.
If $d(\omega) = 0$, then ω is already formed by a (possibly empty) backward path followed by a forward one.
If $d(\omega) > 0$, the result follows by showing that there exists $\omega' \asymp \omega$ with $|\omega'| \leq |\omega|$ and $d(\omega') < d(\omega)$. Since $d(\omega) > 0$, ω contains at least one discording pair. Let the one formed by θ_1 and θ_2 be the earliest one, where $\omega = \overline{\omega_1}\theta_1\theta_1^p\overline{\theta_2}\overline{\theta_2^q}\omega_2$ with ω_1 being forward.
If $\theta_1 = \theta_2$, then trivially $\omega_1\theta_1\theta_1^p\overline{\theta_2}\overline{\theta_2^q}\omega_2 \asymp \overline{\omega_1}\omega_2$ with $|\overline{\omega_1}\omega_2| < |\omega|$ and $d(\overline{\omega_1}\omega_2) < d(\omega)$.
If $\theta_1 \neq \theta_2$ with the two transitions being concurrent, by using the square property (Lemma 1) we can swap them thereby obtaining $\overline{\omega_1}\theta_1\theta_1^p\overline{\theta_2}\overline{\theta_2^q}\omega_2 \asymp \overline{\omega_1}\overline{\theta_2}\overline{\theta_q^2}\theta_1\theta_1^p\omega_2$ with $|\overline{\omega_1}\overline{\theta_2}\overline{\theta_2^q}\theta_1\theta_1^p\omega_2| \leq |\omega|$. If ω_2 starts with a forward transition then $d(\overline{\omega_1}\overline{\theta_2}\theta_1\omega_2) < d(\omega)$, otherwise we keep moving right with θ_1 being

part of the next earliest discording pair to consider.
If $\theta_1 \neq \theta_2$ with the two transitions being in conflict, there is just one case according Definition 1: θ_1 and θ_2 are two transitions with $\overline{\theta_2}$ removing a cause of θ_1 (condition 1). Since it is not possible to perform such a $\overline{\theta_2}$ after θ_1, this case does not apply. ■

Proof of Theorem 1.
It follows from past well foundedness and the parabolic lemma thanks to [29]. ■

B Strictly Alternating RPPC: Syntax and Semantics

The syntax of the standard forward processes for RPPC_{sa}, a variant of RPPC complying with the strictly alternating model of [20], is as follows:

$$\begin{array}{rcl} N, M & ::= & \underline{0} \mid a \,.\, P \mid N + M \mid N \parallel_L M \\ P, Q & ::= & \bigoplus_{h \in H} \langle p_h \rangle N_h \mid P \parallel_L Q \end{array}$$

where H is a finite and non-empty index set, $p_h \in \mathbb{R}_{]0,1]}$ for all $h \in H$, and $\sum_{h \in H} p_h = 1$.

Observing that parallel composition applies to both nondeterministic processes and probabilistic processes, action prefix and nondeterministic choice characterize nondeterministic processes while probabilistic choice characterizes probabilistic processes. The strict alternation arises from the fact that the continuation of an action prefix is a probabilistic process and the continuation of every summand within a probabilistic choice is a nondeterministic process.

The syntax of the corresponding reversible processes is the following:

$$\begin{array}{rcl} RN, RM & ::= & N \mid a[i] \,.\, RP \mid RN + RM \mid RN \parallel_L RM \\ RP, RQ & ::= & P \mid \langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h \mid RP \parallel_L RQ \end{array}$$

where $z \in H$ singles out the summand that has been selected.

The related operational semantic rules are shown in Table 4 for action transitions and in Table 5 for probabilistic transitions. As can be noted, they are fewer than those in Tables 2 and 3, respectively, and also simpler because they use only $\texttt{std}$ and $\texttt{key}_{\texttt{a}}$ and do not need negative premises on probabilistic transitions.

The only subtlety is related to rules PAR_{sa} and $\text{PAR}^{\bullet}_{\text{sa}}$. In the former rule, since RN can do an action and hence is nondeterministic, RM is nondeterministic too and hence has to be made probabilistic in the derivative process, which would simply be accomplished by transforming it into $\langle 1 \rangle RM$ if it were a standard forward process [20]. Function $\texttt{mkpr}$ is inductively defined as follows:

$$\begin{array}{rcl} \texttt{mkpr}(N) & = & \langle 1 \rangle N \\ \texttt{mkpr}(a[i] \,.\, RP) & = & a[i] \,.\, \texttt{mkpr}(RP) \\ \texttt{mkpr}(RN + RM) & = & \begin{cases} \texttt{mkpr}(RN) + RM & \text{if } \texttt{std}(RM) \\ RN + \texttt{mkpr}(RM) & \text{if } \texttt{std}(RN) \end{cases} \\ \texttt{mkpr}(RN \parallel_L RM) & = & \texttt{mkpr}(RN) \parallel_L \texttt{mkpr}(RM) \\ \texttt{mkpr}(P) & = & P \\ \texttt{mkpr}(\langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h) & = & \langle p_z \rangle^{[i]} \texttt{mkpr}(RN_z) \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h \\ \texttt{mkpr}(RP \parallel_L RQ) & = & \texttt{mkpr}(RP) \parallel_L \texttt{mkpr}(RQ) \end{array}$$

In the latter rule, since RN can undo an action and hence is probabilistic, RM is probabilistic too and hence has to be made nondeterministic in the derivative process. Function $\texttt{mknd}$ is inductively defined as follows:

$$\begin{array}{rcl} \texttt{mknd}(\langle 1 \rangle N) & = & N \\ \texttt{mknd}(\langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h) & = & \langle p_z \rangle^{[i]} \texttt{mknd}(RN_z) \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h \\ \texttt{mknd}(RP \parallel_L RQ) & = & \texttt{mknd}(RP) \parallel_L \texttt{mknd}(RQ) \\ \texttt{mknd}(N) & = & N \\ \texttt{mknd}(a[i] \,.\, RP) & = & a[i] \,.\, \texttt{mknd}(RP) \\ \texttt{mknd}(RN + RM) & = & \begin{cases} \texttt{mknd}(RN) + RM & \text{if } \texttt{std}(RM) \\ RN + \texttt{mknd}(RM) & \text{if } \texttt{std}(RN) \end{cases} \\ \texttt{mknd}(RN \parallel_L RM) & = & \texttt{mknd}(RN) \parallel_L \texttt{mknd}(RM) \end{array}$$

$$(\text{Act1}_{\text{sa}})\ \frac{\texttt{std}(RP)}{a\,.\,RP \xrightarrow{a[i]}_{\texttt{a}} a[i]\,.\,RP}$$

$$(\text{Act1}^{\bullet}_{\text{sa}})\ \frac{\texttt{std}(RP)}{a[i]\,.\,RP \overset{a[i]}{\dashrightarrow}_{\texttt{a}} a\,.\,RP}$$

$$(\text{Act2}_{\text{sa}})\ \frac{RP \xrightarrow{b[j]}_{\texttt{a}} RP' \quad j \neq i}{a[i]\,.\,RP \xrightarrow{b[j]}_{\texttt{a}} a[i]\,.\,RP'}$$

$$(\text{Act2}^{\bullet}_{\text{sa}})\ \frac{RP \overset{b[j]}{\dashrightarrow}_{\texttt{a}} RP' \quad j \neq i}{a[i]\,.\,RP \overset{b[j]}{\dashrightarrow}_{\texttt{a}} a[i]\,.\,RP'}$$

$$(\text{Act3}_{\text{sa}})\ \frac{RN_z \xrightarrow{b[j]}_{\texttt{a}} RN'_z \quad z \in H \quad \forall h \in H \setminus \{z\}.\,\texttt{std}(RN_h)}{\langle p_z\rangle^{[i]} RN_z \bigoplus_{h\in H\setminus\{z\}} \langle p_h\rangle RN_h \xrightarrow{b[j]}_{\texttt{a}} \langle p_z\rangle^{[i]} RN'_z \bigoplus_{h\in H\setminus\{z\}} \langle p_h\rangle RN_h}$$

$$(\text{Act3}^{\bullet}_{\text{sa}})\ \frac{RN_z \overset{b[j]}{\dashrightarrow}_{\texttt{a}} RN'_z \quad z \in H \quad \forall h \in H \setminus \{z\}.\,\texttt{std}(RN_h)}{\langle p_z\rangle^{[i]} RN_z \bigoplus_{h\in H\setminus\{z\}} \langle p_h\rangle RN_h \overset{b[j]}{\dashrightarrow}_{\texttt{a}} \langle p_z\rangle^{[i]} RN'_z \bigoplus_{h\in H\setminus\{z\}} \langle p_h\rangle RN_h}$$

$$(\text{Cho}_{\text{sa}})\ \frac{RN \xrightarrow{a[i]}_{\texttt{a}} RN' \quad \texttt{std}(RM)}{RN + RM \xrightarrow{a[i]}_{\texttt{a}} RN' + RM}$$

$$(\text{Cho}^{\bullet}_{\text{sa}})\ \frac{RN \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RN' \quad \texttt{std}(RM)}{RN + RM \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RN' + RM}$$

$$(\text{Par}_{\text{sa}})\ \frac{RN \xrightarrow{a[i]}_{\texttt{a}} RN' \quad a \notin L \quad i \notin \texttt{key}_{\texttt{a}}(RM)}{RN \|_L RM \xrightarrow{a[i]}_{\texttt{a}} RN' \|_L \texttt{mkpr}(RM)}$$

$$(\text{Par}^{\bullet}_{\text{sa}})\ \frac{RN \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RN' \quad a \notin L \quad i \notin \texttt{key}_{\texttt{a}}(RM)}{RN \|_L RM \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RN' \|_L \texttt{mknd}(RM)}$$

$$(\text{Coo}_{\text{sa}})\ \frac{RN \xrightarrow{a[i]}_{\texttt{a}} RN' \quad RM \xrightarrow{a[i]}_{\texttt{a}} RM' \quad a \in L}{RN \|_L RM \xrightarrow{a[i]}_{\texttt{a}} RN' \|_L RM'}$$

$$(\text{Coo}^{\bullet}_{\text{sa}})\ \frac{RN \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RN' \quad RM \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RM' \quad a \in L}{RN \|_L RM \overset{a[i]}{\dashrightarrow}_{\texttt{a}} RN' \|_L RM'}$$

Table 4. Operational semantic rules for RPPC_{sa} action transitions

$$(\text{PSEL1}_{\text{sa}})\ \frac{z \in H \quad \forall h \in H.\, \mathtt{std}(RN_h)}{\bigoplus_{h \in H} \langle p_h \rangle RN_h \xrightarrow{(p_z)^{[i]}}_{\mathtt{p}} \langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h}$$

$$(\text{PSEL1}^{\bullet}_{\text{sa}})\ \frac{z \in H \quad \forall h \in H.\, \mathtt{std}(RN_h)}{\langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h \overset{(p_z)^{[i]}}{\dashrightarrow}_{\mathtt{p}} \bigoplus_{h \in H} \langle p_h \rangle RN_h}$$

$$(\text{PSEL2}_{\text{sa}})\ \frac{RN_z \xrightarrow{(q)^{[j]}}_{\mathtt{p}} RN'_z \quad z \in H \quad \forall h \in H \setminus \{z\}.\, \mathtt{std}(RN_h) \quad j \neq i}{\langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h \xrightarrow{(q)^{[j]}}_{\mathtt{p}} \langle p_z \rangle^{[i]} RN'_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h}$$

$$(\text{PSEL2}^{\bullet}_{\text{sa}})\ \frac{RN_z \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} RN'_z \quad z \in H \quad \forall h \in H \setminus \{z\}.\, \mathtt{std}(RN_h) \quad j \neq i}{\langle p_z \rangle^{[i]} RN_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} \langle p_z \rangle^{[i]} RN'_z \bigoplus_{h \in H \setminus \{z\}} \langle p_h \rangle RN_h}$$

$$(\text{PSEL3}_{\text{sa}})\ \frac{RP \xrightarrow{(q)^{[j]}}_{\mathtt{p}} RP'}{a[i]\,.\,RP \xrightarrow{(q)^{[j]}}_{\mathtt{p}} a[i]\,.\,RP'}$$

$$(\text{PSEL3}^{\bullet}_{\text{sa}})\ \frac{RP \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} RP'}{a[i]\,.\,RP \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} a[i]\,.\,RP'}$$

$$(\text{PSEL4}_{\text{sa}})\ \frac{RN \xrightarrow{(q)^{[j]}}_{\mathtt{p}} RN' \quad \mathtt{std}(RM)}{RN + RM \xrightarrow{(q)^{[j]}}_{\mathtt{p}} RN' + RM}$$

$$(\text{PSEL4}^{\bullet}_{\text{sa}})\ \frac{RN \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} RN' \quad \mathtt{std}(RM)}{RN + RM \overset{(q)^{[j]}}{\dashrightarrow}_{\mathtt{p}} RN' + RM}$$

$$(\text{PCOO}_{\text{sa}})\ \frac{RP \xrightarrow{(p)^{[i]}}_{\mathtt{p}} RP' \quad RQ \xrightarrow{(q)^{[i]}}_{\mathtt{p}} RQ'}{RP \|_L RQ \xrightarrow{(p \cdot q)^{[i]}}_{\mathtt{p}} RP' \|_L RQ'}$$

$$(\text{PCOO}^{\bullet}_{\text{sa}})\ \frac{RP \overset{(p)^{[i]}}{\dashrightarrow}_{\mathtt{p}} RP' \quad RQ \overset{(q)^{[i]}}{\dashrightarrow}_{\mathtt{p}} RQ'}{RP \|_L RQ \overset{(p \cdot q)^{[i]}}{\dashrightarrow}_{\mathtt{p}} RP' \|_L RQ'}$$

Table 5. Operational semantic rules for RPPC_{sa} probabilistic transitions