

Reversing P/T Nets [★]

Hernán Melgratti¹, Claudio Antares Mezzina^{2,3}, and Irek Ulidowski²

¹ University of Buenos Aires - Conicet, Argentina

² University of Leicester, United Kingdom

³ Dipartimento di Scienze Pure e Applicate, Università di Urbino, Italy

Abstract. Petri Nets are a well-known model of concurrency and provide an ideal setting for the study of fundamental aspects in concurrent systems. Despite their simplicity, they still lack a satisfactory causally reversible semantics. We develop such semantics for Place/Transitions Petri Nets (P/T nets) based on two observations. Firstly, a net that explicitly expresses causality and conflict among events, e.g., an occurrence net, can be straightforwardly reversed by adding reversal for each of its transitions. Secondly, the standard unfolding construction associates a P/T net with an occurrence net that preserves all of its computation. Consequently, the reversible semantics of a P/T net can be obtained as the reversible semantics of its unfolding. We show that such reversible behaviour can be expressed as a finite net whose tokens are coloured by causal histories. Colours in our encoding resemble the causal memories that are typical in reversible process calculi.

1 Introduction

Reversible computing is attracting interest for its applications in many fields including hardware design and quantum computing [27], the modelling of biochemical reactions [23,11,24], parallel discrete event simulation [25] and program reversing for debugging [7,14,10].

A model for reversible computation features two computation flows: the standard forward direction and the reverse one, which allows to reach back any past state of the computation. Reversibility is well understood in a sequential setting in which executions are totally ordered sets of events (see [15]): a sequential computation can be reversed by successively undoing the last not yet undone event. Reversibility becomes more challenging in a concurrent setting because there is no natural way for totally ordering events. Often concurrency models account for the causal dependencies among events, which are reflected as a partial order. Reversing an execution consisting of a partially ordered set of events reduces to successively undoing one of the maximal events not yet undone. This is at the basis of the *causally-consistent reversibility* [6,21,13], which relates reversibility

[★] Research partly supported by the EU H2020 RISE programme under the Marie Skłodowska-Curie grant agreement No 778233. The second author acknowledge the full support from the Marie Skłodowska-Curie Individual Fellowship RCADE No 794405.

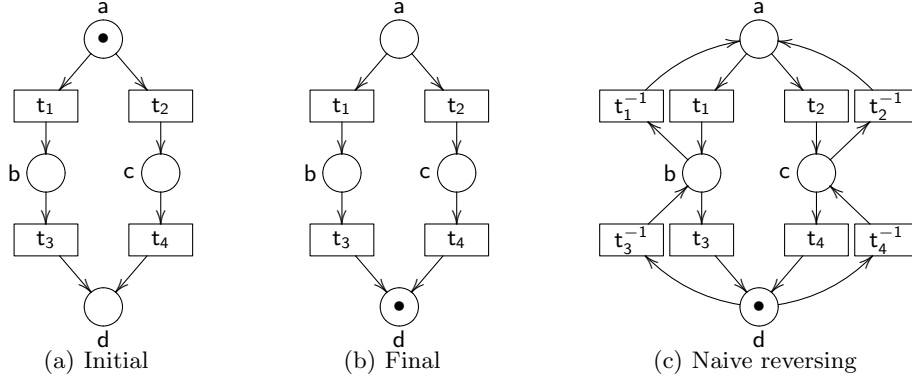


Fig. 1. Backward conflict and naive reversing.

with causality. Intuitively, this notion stipulates that any event can be undone provided that all its consequences, if any, are undone beforehand. The interplay between reversibility and concurrency has been widely studied in process calculi [6,21,12,4,17], event structures [22,5,26,8] and lately Petri Nets [19,1]. Despite being a very basic model of concurrency, Petri nets still lack a satisfactory causally-consistent reversible semantics. For instance, no current models are able to handle cyclic nets.

A key point when reversing computation in Petri nets is to handle backward conflicts, i.e., the fact that a token can be generated in a place because of different causes. Consider the net in Figure 1(a) showing the initial state of a system that can either perform t_1 followed by t_3 , or t_2 followed by t_4 . The final state of a complete computation is depicted in Figure 1(b). The information in that state is not enough to deduce whether the token in d has been produced because of t_3 or t_4 . Even worse, if we “naively” reverse the net by just adding transitions in the reverse direction, as shown in Figure 1(c), the reverse transition will do more than undoing the computation. In fact, the token in d can be put back either in b or c regardless of the previous computation.

Analogous problems arise when a net is cyclic. Previous approaches [19,1] to reversing Petri nets tackle backward conflicts by relying on a new kind of tokens, called *bonds* that keep track of the execution history. Bonds are rich enough for allowing other approaches to reversibility, such as *out-of-order* reversibility [11], but they cannot cope with cyclic nets. We propose here a reversible model for P/T nets that can handle cyclic nets by relying on standard notions in Petri net theory. We first observe that a Petri Net can be mapped via the standard unfolding construction to an occurrence net, i.e., an acyclic net that does not have backward conflicts and makes causal dependencies explicit. Then, an occurrence net can be “simply” reversed by reversing each of its transitions. Such construction gives a model that features causally-consistent reversibility. This is shown by proving that each reachable marking in the reversible version of the

occurrence net is a marking that can be reached by just forward computational steps. We observe that the unfolding construction could produce an infinite occurrence net. However, the unfolding can be seen as the definition of a coloured net, where colours account for causal histories. Such interpretation associates a P/T net with an equivalent coloured P/T net, which can be reversed in the “simple” way. The correctness of the construction is shown by exhibiting a one-to-one correspondence of its executions with the ones of the reversible version of the unfolding. Interestingly, the colours used by the construction resemble the memories common in reversible calculi [12,6].

We remark that our proposal deals with reversing (undoing) computation in a Petri net and not with the classical problem of reversibility [3] which requires every computation to be able to reach back the initial state of the system (but not necessary by undoing the previous events). In this sense, the problem of making a net reversible equates to adding a minimal amount of transitions that make a net reversible [2]. Reversibility is a global property while reversing a computation is a local one, as discussed in [2].

2 Background

2.1 Petri Nets

Petri nets are built up from *places* (denoting, e.g., resources and message types), which are repositories for *tokens* (representing instances of resources), and *transitions*, which fetch and produce tokens. We consider the infinite sets \mathcal{P} of places and \mathcal{T} of transitions, and assume that they are disjoint, i.e., $\mathcal{P} \cap \mathcal{T} = \emptyset$. We let a, a', \dots range over \mathcal{P} and t, t', \dots over \mathcal{T} . We write x, y, \dots for elements in $\mathcal{P} \cup \mathcal{T}$.

A *multiset* over a set S is a function $m : S \rightarrow \mathbb{N}$ (where \mathbb{N} denotes the natural numbers including zero). We write \mathbb{N}^S for the set of multisets over S . For $m \in \mathbb{N}^S$, $\text{supp}(m) = \{x \in S \mid m(x) > 0\}$ is the *support* of m , and $|m| = \sum_{x \in S} m(x)$ stands for its *cardinality*. We write \emptyset for the empty multiset, i.e., $\text{supp}(\emptyset) = \emptyset$. The union of $m_1, m_2 \in \mathbb{N}^S$, written $(m_1 \oplus m_2)$, is defined such that $(m_1 \oplus m_2)(x) = m_1(x) + m_2(x)$ for all $x \in S$. Note that \oplus is associative and commutative, and has \emptyset as identity. Hence, \mathbb{N}^S is the free commutative monoid S^\oplus over S . We write x for a singleton multiset, i.e., $\text{supp}(x) = \{x\}$ and $m(x) = 1$. Moreover, we write $x_1 \dots x_n$ for $x_1 \oplus \dots \oplus x_n$. Let $f : S \rightarrow S'$, we write f also for its obvious extension to multisets, i.e., $f(x_0 \dots x_n) = f(x_0) \dots f(x_n)$. We avoid writing $\text{supp}(-)$ when applying set operators to multisets, e.g., we write $x \in m$ or $m_1 \cap m_2$ instead of $x \in \text{supp}(m)$ or $\text{supp}(m_1) \cap \text{supp}(m_2)$.

Definition 1 (Petri Net). A net N is a 4-tuple $N = (S_N, T_N, \bullet_{-N}, \bullet_N)$ where $S_N \subseteq \mathcal{P}$ is the (nonempty) set of places, $T_N \subseteq \mathcal{T}$ is the set of transitions and the functions $\bullet_{-N}, \bullet_N : T_N \rightarrow 2^{S_N}$ assign source and target to each transition such that $\bullet_t \neq \emptyset$ and $\bullet_t \neq \emptyset$ for all $t \in T_N$. A marking of a net N is a multiset over S_N , i.e., $m \in \mathbb{N}^{S_N}$. A Petri net is a pair (N, m) where N is a net and m is a marking of N .

We denote $S_N \cup T_N$ by N , and omit the subscript N if no confusion arises. We abbreviate a transition $t \in T$ with *preset* $\bullet t = s_1$ and *postset* $t \bullet = s_2$ as $s_1 \rceil s_2$. Hereafter, we only consider nets whose transitions have non-empty presets. The pre and postset of a place $a \in S$ are defined respectively as $\bullet a = \{t \mid a \in t \bullet\}$ and $a \bullet = \{t \mid a \in \bullet t\}$. We let ${}^\circ N = \{x \in N \mid \bullet x = \emptyset\}$ and $N^\circ = \{x \in N \mid x \bullet = \emptyset\}$ denote the sets of *initial* and *final elements* of N respectively. Note that we only consider nets whose initial and final elements are places since transitions have non-empty pre and postsets, i.e., $\bullet t \neq \emptyset$ and $t \bullet \neq \emptyset$ holds for all t .

Definition 2 (Net morphisms). *Let N, N' be nets. A pair $f = (f_S : S_N \rightarrow S_{N'}, f_T : T_N \rightarrow T_{N'})$ is a net morphism from N to N' (written $f : N \rightarrow N'$) if $f_S(\bullet t_N) = \bullet(f_T(t))_{N'}$ and $f_S(t \bullet_N) = (f_T(t))_{N'}^\bullet$ for any t . Moreover, we say N and N' are isomorphic if f is bijective.*

The operational (interleaving) semantics of a Petri net is given by the least relation on Petri nets satisfying the following inference rule:

$$\frac{\text{(FIRING)} \quad t = m \rceil m' \in T_N}{(N, m \oplus m'') \xrightarrow{t} (N, m' \oplus m'')}$$

which describes the evolution of the state of a net (represented by the marking $m \oplus m''$) by the firing of a transition $m \rceil m'$ that consumes the tokens m in its preset and produces the tokens m' in its postset. We sometimes omit t in \xrightarrow{t} when the fired transition is uninteresting.

According to Definition 1, transitions consume and produce at most one token in each place. On the other hand, P/T nets below fetch and consume multiple tokens by defining the pre- and postsets of transitions as multisets.

Definition 3 (P/T net). *A Place / Transition Petri net (P/T net) is a 4-tuple $N = (S_N, T_N, \bullet_{-N}, \bullet_{-N})$ where $S_N \subseteq \mathcal{P}$ is the (nonempty) set of places, $T_N \subseteq \mathcal{T}$ is the set of transitions and the functions $\bullet_{-N}, \bullet_{-N} : T_N \rightarrow \mathbb{N}^{S_N}$ assign source and target to each transition. A marking of a net N is multiset over S_N , i.e., $m \in \mathbb{N}^{S_N}$. A marked P/T net is a pair (N, m) where N is a P/T net and m is a marking of N .*

The notions of pre- and postset, initial and final elements, morphisms and operational semantics are straightforwardly extended to P/T nets. Note that Petri nets can be regarded as a P/T net whose arcs have unary weights.

Next, we introduce some notation for sequences of transitions. Let ‘;’ denote concatenation of such sequences. For the sequence $s = t_1; t_2; \dots; t_n$, we write $(N, m_0) \xrightarrow{s} (N, m_n)$ if $(N, m_0) \xrightarrow{t_1} (N, m_1) \xrightarrow{t_2} \dots \xrightarrow{t_n} (N, m_n)$; we call s a firing sequence. We write $(N, m_0) \rightarrow^* (N, m_n)$ if there exists s such that $(N, m_0) \xrightarrow{s} (N, m_n)$, and ϵ_m for the empty sequence.

Definition 4. *Let (N, m) be a P/T net. The set of reachable markings $\text{reach}(N, m)$ is defined as $\{m' \mid (N, m) \rightarrow^* (N, m')\}$.*

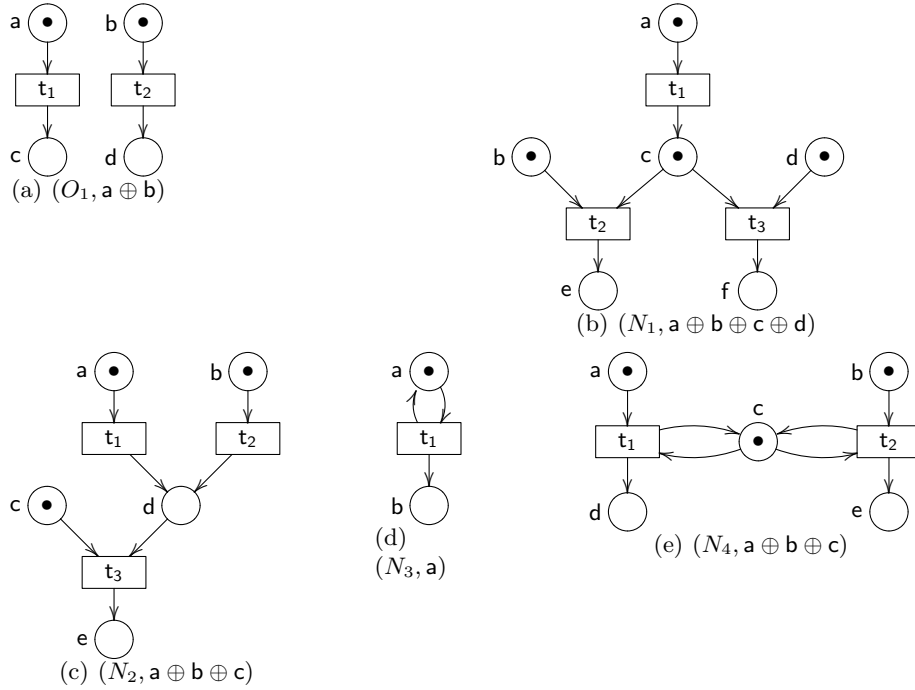


Fig. 2. P/T nets

We say a marked P/T net (N, m) is *(1-)safe* if every reachable marking is a set, i.e., $m' \in \text{reach}(N, m)$ implies $m' \in 2^{S_N}$.

Example 1. Figure 2 shows different P/T nets, which will be used throughout the paper. As usual, places and transitions are represented by circles and boxes, respectively. The nets O_1 and N_4 are Petri nets, and N_1 , N_2 and N_3 are P/T nets which, when executing, may produce multiple tokens in some places.

2.2 Unfolding of P/T nets

Our approach to reversing Petri nets relies on their occurrence net semantics, which explicitly exhibit the causal ordering, concurrency, and conflicts among events. We start by introducing several useful notions and notations. First, we shall describe a flow of causal dependencies in a net with the relation \prec :

Definition 5. Let \prec be $\{(a, t) \mid a \in S_N \wedge t \in a^\bullet\} \cup \{(t, a) \mid a \in S_N \wedge t \in {}^\bullet a\}$. We write \preceq for the reflexive and transitive closure of \prec .

Consider Figure 2. We have $a \prec t_1$ and $t_1 \prec c$ in O_1 as well as $t_1 \preceq t_2$ in N_1 .

Two transitions t_1 and t_2 are in an *immediate conflict*, written $t_1 \#_0 t_2$, when $t_1 \neq t_2$ and ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$. For example, t_1 and t_2 in N_4 in Figure 2 are in an immediate conflict since they share a token in the place c . Correspondingly, for

t_2 and t_3 in N_1 . The *conflict* relation $\#$ is defined by letting $x\#y$ if $x \neq y$ and there are $t_1, t_2 \in T$ such that $t_1 \preceq x$, and $t_2 \preceq y$, and $t_1 \#_0 t_2$.

We are now ready to give the definition of an occurrence net following [18,9].

Definition 6 (Occurrence net). A net (N, m) is an occurrence net if

1. N is acyclic;
2. N is a (1-)safe net, i.e., any reachable marking is a set;
3. $m = {}^\circ N$, i.e., the initial marking is identified with the set of initial places;
4. there are no backward conflicts, i.e., $|\bullet a| \leq 1$ for all a in S_N ;
5. there are no self-conflicts, i.e., $\neg(t\#t)$ for all t in T_N .

We use O to range over occurrence nets.

Example 2. The net O_1 in Figure 2 is an occurrence net, while the remaining nets are not. N_1 is not an occurrence net since there is a token in place c and c is not an initial place of the net. N_2 has a backward conflict since two transitions produce tokens on the place d . N_3 is cyclic, and N_4 is cyclic and has a backward conflict on c .

The absence of backward conflicts in occurrence nets ensures that each place appears in the postset of at most one transition. Hence, pre- and postset relations can be interpreted as a causal dependency. So, \preceq represents causality.

We say $x, y \in N$ are *concurrent*, written $x \text{ co } y$, if $x \neq y$ and $x \not\preceq y$, $y \not\preceq x$, and $\neg x\#y$. A set $X \subseteq N$ is concurrent, written $CO(X)$, if $\forall x, y \in X : x \neq y \Rightarrow x \text{ co } y$, and $|\{t \in T_N \mid \exists x \in X, t \preceq x\}|$ is finite. For example, the set $\{t_1, t_2\}$ of firings in O_1 of Figure 2 is concurrent, so we can write $CO(\{t_1, t_2\})$.

Two transitions are *coinitial* if they start with the same marking, and *cofinal* if they end up in the same marking. We now have a simple version of the Square Lemma [6] for forward concurrent transitions. It will be helpful in proving our Lemma 4 in the next section.

Lemma 1. *Let t and t' be coinitial concurrent transitions. Then, there exist transitions t_1 and t'_1 such that $t; t'_1$ and $t'; t_1$ are cofinal.*

The lemma says that if transitions t and t' originate from one corner of a square, and if they represent independent (concurrent) events, then the square completes with two other independent transitions (t_1 and t'_1) meeting at the opposite corner of the square. The order in which concurrent transitions are executed in a firing sequence does not matter. Indeed, the order which should be preserved among firings in a sequence is the causal order. We then consider sequences equivalent up to the swapping of concurrent transitions. This corresponds to considering the set of Mazurkiewicz traces induced by co as the independence relation.

Formally, trace equivalence \equiv is the least congruence over firing sequences s such that $\forall t_1, t_2 : t_1 \text{ co } t_2 \implies t_1; t_2 \equiv t_2; t_1$. The equivalence classes of \equiv are the (Mazurkiewicz) *traces*. We use ω to range over such traces. We also will use ϵ for the empty trace, and $;$ for the concatenation operator.

$$\begin{array}{c}
\text{(INI-MK)} \\
\hline
m(\mathbf{a}) = n \\
\hline
\{\mathbf{a}(\emptyset, i) \mid 1 \leq i \leq n\} \subseteq S \\
\\
\text{(PRE)} \\
\hline
H = \{\mathbf{a}_j(h_j, i_j) \mid j \in J\} \subseteq S \quad Co(H) \quad \mathbf{t} \in T_N \quad \bullet \mathbf{t}_N = \oplus_{j \in J} \mathbf{a}_j \\
\hline
\mathbf{t}(H) \in T, \quad \bullet(\mathbf{t}(H)) = H \\
\\
\text{(POST)} \\
\hline
x = \mathbf{t}(H) \in T \\
\hline
Q = \{\mathbf{a}(\{x\}, i) \mid 1 \leq i \leq \bullet_N(\mathbf{a})\} \subseteq S, \quad x^\bullet = Q
\end{array}$$

Fig. 3. Unfolding rules.

For occurrence nets we have this standard property:

$$s_1 \equiv s_2 \text{ iff } (O, m_0) \xrightarrow{s_1} (O, m_n) \iff (O, m_0) \xrightarrow{s_2} (O, m_n) \quad (1)$$

Two traces are *coinitial* if they start with the same marking, and *cofinal* if they end up in the same marking. Hence, equation (1) tells us that two traces that are *coinitial* and *cofinal* are then trace equivalent.

The unfolding of a net N is the least occurrence net that can account for all the possible computations of N and makes explicit causal dependencies, conflicts and concurrency between firings [18].

Definition 7 (Unfolding). Let (N, m) be a P/T net. The unfolding of N is the occurrence net $\mathcal{U}[N, m] = (S, T, \delta_0, \delta_1)$ generated inductively by the inference rules in Figure 3 and the folding morphism $(f_S, f_T) : \mathcal{U}[N, m] \rightarrow N$ defined such that $f_S(\mathbf{a}, -, -) = \mathbf{a}$ and $f_T(\mathbf{t}, -) = \mathbf{t}$.

Places are named by triples $\mathbf{a}(H, i)$ where: \mathbf{a} is a place of N where tokens reside; H is the set of immediate causes (i.e., the history of tokens); and i is a positive integer used to disambiguate tokens with the same history. Transitions (or events) are encoded as $\mathbf{t}(H)$, where H is as above and \mathbf{t} is the fired transition.

Example 3. The unfoldings of the nets $(N_1, \mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c} \oplus \mathbf{d})$, $(N_2, \mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c})$ and (N_3, \mathbf{a}) in Figure 2 are shown in Figure 4. Note that since O_1 is an occurrence net its unfolding is isomorphic to O_1 , thus it is omitted. Consider the occurrence net $\mathcal{U}[N_1, \mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c} \oplus \mathbf{d}]$. The leftmost transition \mathbf{t}_2 is different from the other transition \mathbf{t}_2 since they have different histories: the leftmost \mathbf{t}_2 is caused by the tokens in \mathbf{b} and \mathbf{c} (which are available in the initial marking), whereas the other \mathbf{t}_2 is caused only by the token in \mathbf{b} and the token that is produced by the firing of \mathbf{t}_1 . Correspondingly, for the two transitions labelled \mathbf{t}_3 . Consider $\mathcal{U}[N_2, \mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c}]$. After the transitions \mathbf{t}_1 and \mathbf{t}_2 have fired, there is a token in each of the places labelled \mathbf{d} . The token in the leftmost \mathbf{d} has the history \mathbf{t}_1 and the token in the other \mathbf{d} has the history \mathbf{t}_2 . Once \mathbf{t}_3 has fired, we can tell the copies of \mathbf{t}_3 apart by inspecting their histories: the leftmost \mathbf{t}_3 is caused by a token in \mathbf{d} with the history \mathbf{t}_1 (as well as the token in \mathbf{c}), whereas the other \mathbf{t}_3 is caused by \mathbf{d} with the history \mathbf{t}_2 and by \mathbf{c} .

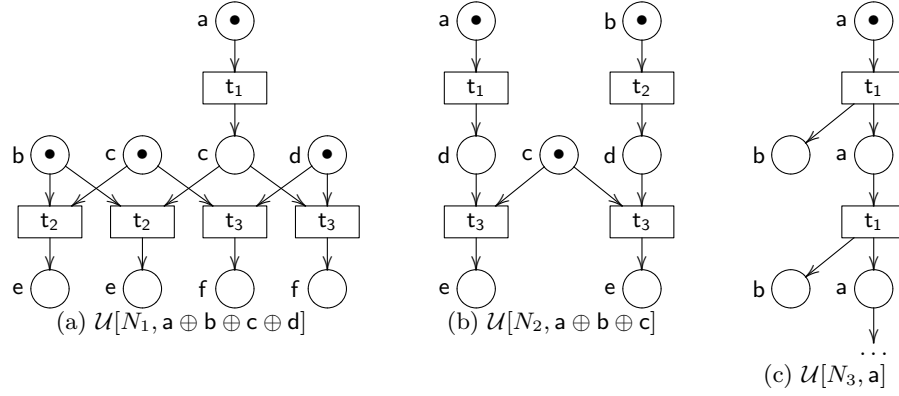


Fig. 4. Unfoldings of P/T nets

3 Reversing Occurrence Nets

Definition 8. Let O be an occurrence net. The reversible version of O is $\overleftarrow{O} = (S_{\overleftarrow{O}}, T_{\overleftarrow{O}}, \bullet_{\overleftarrow{O}}, \circ_{\overleftarrow{O}})$ defined such that

$$S_{\overleftarrow{O}} = S_O \quad T_{\overleftarrow{O}} = T_O \cup \{ \overleftarrow{t} \mid t \in T_O \}$$

$$\bullet_{\overleftarrow{O}} = \begin{cases} \bullet_{t_O} & \text{if } t \in T_O \\ \bullet_{\overleftarrow{t}_O} & \text{otherwise} \end{cases} \quad \circ_{\overleftarrow{O}} = \begin{cases} \circ_{t_O} & \text{if } t \in T_O \\ \circ_{\overleftarrow{t}_O} & \text{otherwise} \end{cases}$$

Given a transition t we write \overleftarrow{t} for a transition that reverses t . We shall call transitions like \overleftarrow{t}_1 and \overleftarrow{t}_2 in Figure 5 *reverse* (or backwards) transitions (or firings), and use t, t_1 and t_2 to denote transitions or reverse transitions.

For \overleftarrow{O} , we write $(\overleftarrow{O}, m) \xrightarrow{t} (\overleftarrow{O}, m')$ for a forward firing when $t \in T_O$, and $(\overleftarrow{O}, m) \xrightarrow{\overleftarrow{t}} (\overleftarrow{O}, m')$ for the reverse (or backward) firing when $t \notin T_O$. We also let \xrightarrow{t} be $\xrightarrow{t} \cup \xrightarrow{\overleftarrow{t}}$. We will often refer to a firing $(\overleftarrow{O}, m) \xrightarrow{t} (\overleftarrow{O}, m')$ as t . Given a firing t we indicate with \overleftarrow{t} its inverse that is

$$(\overleftarrow{O}, m) \xrightarrow{\overleftarrow{t}} (\overleftarrow{O}, m') \quad \text{if} \quad (\overleftarrow{O}, m') \xrightarrow{t} (\overleftarrow{O}, m)$$

$$(\overleftarrow{O}, m) \xrightarrow{t} (\overleftarrow{O}, m') \quad \text{if} \quad (\overleftarrow{O}, m') \xrightarrow{\overleftarrow{t}} (\overleftarrow{O}, m)$$

Hence, we have $\overleftarrow{\overleftarrow{t}} = t$. We shall work with sequences of transitions and reverse transitions, ranged over by s, s_1 and s_2 . We say that a sequence is a *forward* (resp. *backward*) *sequence* when all its firings are forward (resp. backward).

Next, we extend the notions of causality, conflict and concurrency to transitions and reverse transitions in reverse versions of occurrence nets. We extend \prec in Definition 5 to cover reverse transitions in an obvious way using Definition 8.

As a result, we obtain $t \preceq \overleftarrow{t}$ and $\overleftarrow{t} \preceq t$. As for the conflict relation, we define an immediate conflict between different $\overleftarrow{t_1}$ and $\overleftarrow{t_2}$ as $\bullet \overleftarrow{t_1} \cap \bullet \overleftarrow{t_2} \neq \emptyset$. This is $t_1 \bullet \cap t_2 \bullet \neq \emptyset$, meaning t_1 and t_2 are in backward conflict, which is ruled out in occurrence nets. Hence, the immediate conflict relation is empty between reverse transitions, and so is the conflict relation. The immediate conflict relation between t and $\overleftarrow{t'}$ is defined as $\bullet t \cap \bullet \overleftarrow{t'} \neq \emptyset$. This is equivalent to $\bullet t \cap t' \bullet \neq \emptyset$, which means $t' \preceq t$. Consequently, the conflict relation on transitions in \overleftarrow{O} is given by the conflict relation on the forward transitions, and can be defined using the causality relation for pairs of a transition and reverse transition. This allows us to define concurrent transitions in \overleftarrow{O} . We say $t \text{ co } t'$ if (a) $t \text{ co } t'$ for $t, t' \in T_O$, (b) $t \not\preceq t'$ and $t' \not\preceq t$ if t, t' are reverse transitions, and (c) $t \not\preceq t', t' \not\preceq t$ and $\overleftarrow{t'} \not\preceq t$ if t is a transition and t' is a reverse transition.

Next, we show that \overleftarrow{O} is a conservative extension of O .

Lemma 2. $(O, m) \xrightarrow{t} (O, m') \text{ iff } (\overleftarrow{O}, m) \xrightarrow{t} (\overleftarrow{O}, m')$.

In general, a reversible occurrence net is not an occurrence net. This is because adding reverse transitions may introduce backward conflict for these transitions. Consider N_1 in Figure 2. We notice that initially t_1 and t_2 are in conflict. Then, in \overleftarrow{N}_1 in Figure 5, the place c with a token has two reverse transitions in its preset, namely $\overleftarrow{t_2}$ and $\overleftarrow{t_3}$, hence there is a backward conflict.

4 Properties

We now study the properties of the reversible versions of occurrence nets.

An important property of a *fully* reversible system is the Loop Lemma stating that any reduction can be undone. Formally:

Lemma 3 (Loop Lemma). $(\overleftarrow{O}, m) \xrightarrow{t} (\overleftarrow{O}, m') \text{ iff } (\overleftarrow{O}, m') \xrightarrow{\overleftarrow{t}} (\overleftarrow{O}, m)$.

We can generalise the result of the Loop Lemma to sequences as follows:

Corollary 1. $(\overleftarrow{O}, m) \rightarrow^* (\overleftarrow{O}, m') \text{ iff } (\overleftarrow{O}, m') \rightarrow^* (\overleftarrow{O}, m)$.

Next, we have a lemma which is instrumental for the proof of causal-consistent reversibility in reversible calculi [6,12]. Note that t and t' can be either forward or reverse transitions.

Lemma 4 (Square Lemma). *Let t and t' be coinital concurrent transitions. Then, there exist transitions t_1 and t'_1 such that $t; t'_1$ and $t'; t_1$ are cofinal.*

In order to prove causal consistency we first define a notion of equivalence on sequences of transitions and reverse transitions in reversible occurrence nets. By following Lévy's approach [16], we define the notion of *reverse equivalence* on such sequences as the least equivalence relation \asymp which is closed under

composition with $;$ such that the following hold (recall that t, t' are transitions or reverse transitions):

$$t; t' \asymp t'; t \quad \text{if } t \text{ co } t' \qquad t; \overleftarrow{t} \asymp \epsilon \qquad \overleftarrow{t}; t \asymp \epsilon$$

Reversible equivalence \asymp allows us to swap the order of t and t' in an execution sequence as long as t, t' are concurrent. Moreover, it allows cancellation of a transition and its inverse. We have that $\equiv \subset \asymp$. The equivalence classes of \asymp are called *traces*; it is clear that they contain the Mazurkiewicz traces. Hence, we shall use ω, ω_1 and ω_2 to range over such traces.

The following lemma says that, up to reverse equivalence, one can always reach for the maximum freedom of choice, going backward, and only then going forwards.

Lemma 5 (Parabolic Lemma). *Let ω be a trace. There exist two forward traces ω_1 and ω_2 such that $\omega \asymp \overleftarrow{\omega_1}; \omega_2$.*

Proof. By lexicographic induction on length of ω and on the distance between the beginning of ω and the earliest pair of opposing firings in ω . The analysis uses both the Loop Lemma (Lemma 3) and the Square Lemma (Lemma 4).

The following lemma says that, if two traces ω_1 and ω_2 are coinitial and cofinal (e.g. they start from the same marking and end in the same marking) and ω_2 is a forward only trace, then ω_1 has some forward firings and their reverse ones that cancel each other. And this implies that ω_1 is causally equivalent to a forward trace in which all those pairs of fairing are cancelled out.

Lemma 6 (Shortening Lemma). *Let $\omega_1 \asymp \omega_2$ with ω_2 forward. Then, $|\omega_2| \leq |\omega_1|$.*

Proof. The proof is by induction on length of ω_1 , using Lemma 4 and Lemma 5. In the proof, the forward trace ω_2 is the main guideline for shortening ω_1 into a forward trace. Indeed, the proof relies crucially on the fact that ω_1 and ω_2 share the same source and target and that ω_2 is a forward trace.

Theorem 1 (Causal Consistency). *Two traces ω_1 and ω_2 are reversible equivalent iff they are coinitial and cofinal, namely*

$$\omega_1 \asymp \omega_2 \text{ iff } (\overleftarrow{O}, m_0) \xrightarrow{\omega_1} (\overleftarrow{O}, m_n) \iff (\overleftarrow{O}, m_0) \xrightarrow{\omega_2} (\overleftarrow{O}, m_n).$$

Proof. The “if” direction follows by definition of reverse equivalence and trace composition. The “only if” direction exploits the properties the Square, Parabolic and Shortening Lemmas.

With Theorem 1 we proved that the notion of causal consistency characterises a space for admissible rollbacks which are: (1) consistent (in the sense that they do not lead to previously unreachable configurations) and (2) flexible enough to allow rearranging of undo actions. This implies that starting from an initial marking, all the markings reached by mixed computations are markings that could be reached by performing only forward computations. Hence, we have:

Theorem 2. *Let O be an occurrence net and m_0 an initial marking. Then,*

$$(\overleftarrow{O}, m_0) \rightarrow^* (\overleftarrow{O}, m') \iff (\overleftarrow{O}, m_0) \twoheadrightarrow^* (\overleftarrow{O}, m').$$

5 Reversing P/T nets

This section takes advantage of the classical unfolding construction for P/T nets and the reversible semantics of occurrence nets to add causally-consistent reversibility to P/T nets.

Definition 9. *Let (N, m) be a marked P/T net and $\mathcal{U}[N, m]$ its unfolding. The reversible version of (N, m) , written $\overleftarrow{(N, m)}$, is $\overleftarrow{\mathcal{U}[N, m]}$.*

Example 4. The reversible version of the nets in Figure 2 are shown in Figure 5. We remark that they are the reversible versions of the nets in Figure 4, which are the unfoldings of the original nets.

The following result states that a reversible net is a conservative extension of its original version, i.e., reversibility does not change the set of reachable markings. The result is a direct consequence of Lemma 2 and the fact that unfoldings preserve reductions up-to the folding morphism \mathcal{U} .

Lemma 7. *$(N, m) \rightarrow^* (N, m')$ iff $\overleftarrow{(N, m)} \twoheadrightarrow^* (\overleftarrow{O}, m'')$ and $m' = f_s(m'')$, where $(f_s, f_t) : \mathcal{U}[N, m] \rightarrow N$, defined such that $f_s(a, -, -) = a$ and $f_t(t, -) = t$, is the folding morphism.*

We remark that the reversible version of a P/T is defined as the reversible version of an occurrence net (i.e., its unfolding). Consequently, all properties shown in the previous section apply to the reversible semantics of P/T nets. In particular, Lemma 7 combined with Theorem 2 ensures that all markings reachable by the reversible semantics are just the reachable markings of the original P/T net.

6 Finite Representation of Reversible P/T Nets

As shown in Figure 5(c), the reversible version of a finite net may be infinite. In this section we show how to represent reversible nets in a compact, finite way by using coloured Petri nets. We assume infinite sets \mathcal{X} of variables and \mathcal{C} of colours, defined such that $\mathcal{X} \subset \mathcal{C}$. For $c \in \mathcal{C}$, we write $\text{vars}(c)$ for the set of variables in c . With abuse of notation we write $\text{vars}(m)$ for the set of variables in a multiset $m \in \mathbb{N}^{\mathcal{P} \times \mathcal{C}}$. Let $\sigma : \mathcal{X} \rightarrow \mathcal{C}$ be a partial function and c a colour (also, $m \in \mathbb{N}^{\mathcal{P} \times \mathcal{C}}$), we write $c\sigma$ (resp., $m\sigma$) for the simultaneous substitution of each variable x in c (resp., m) by $\sigma(x)$.

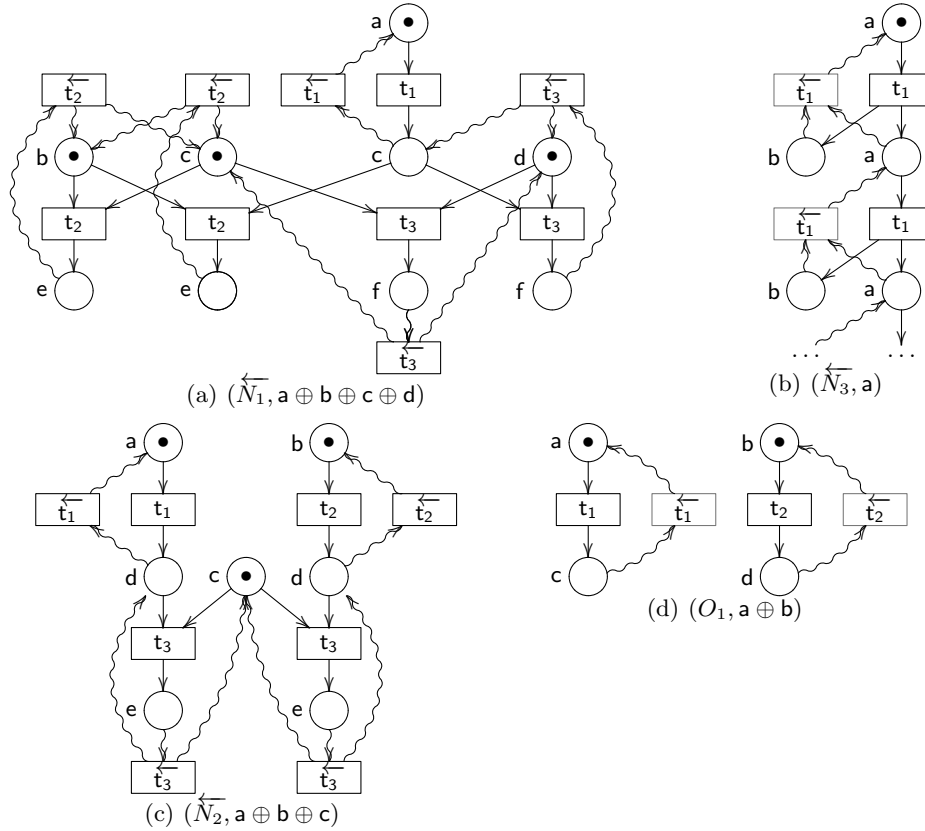


Fig. 5. Reversible P/T and Petri nets

Definition 10 (C-P/T net). A coloured place / transition net (C-P/T net) is a 4-tuple $N = (S_N, T_N, \bullet_{-N}, \bullet_N)$ where $S_N \subseteq \mathcal{P}$ is the (nonempty) set of places, $T_N \subseteq \mathcal{T}$ is the set of transitions and the functions $\bullet_{-N}, \bullet_N : T_N \rightarrow \mathbb{N}^{S_N \times \mathcal{C}}$ assign source and target to each transition defined such that $\text{vars}(\mathbf{t}^\bullet) \subseteq \text{vars}(\bullet \mathbf{t})$. A marking of a C-P/T net N is multiset over $S_N \times \mathcal{C}$ that does not contain variables, i.e., $m \in \mathbb{N}^{S_N \times \mathcal{C}}$ and $\text{vars}(m) = \emptyset$. A marked C-P/T net is a pair (N, m) where N is a P/T net and m is a marking of N .

C-P/T nets generalise P/T nets by extending markings to multisets of coloured tokens, and transitions to patterns that need to be instantiated with appropriate colours for firing, as formally stated by the firing rule below.

$$\begin{array}{c}
 \text{(COLOURED-FIRING)} \\
 \mathbf{t} = m \mid m' \in T_N \\
 \hline
 (N, m\sigma \oplus m'') \xrightarrow{\mathbf{t}} (N, m'\sigma \oplus m'')
 \end{array}$$

The firing of a transition $t = m \rhd m'$ requires to instantiate m and m' by substituting variables by colours, i.e., the firing of t consumes the instance $m\sigma$ of the preset m and produces the instance $m'\sigma$ of the postset of m' .

We now introduce an encoding that associates each P/T net N with an equivalent C-P/T net $\llbracket N \rrbracket$, whose tokens carry their execution history. We rely on the set of colours \mathcal{C} defined as the least set that contains \mathcal{X} and it is closed under the following rules.

$$\frac{\text{(TOKEN)} \quad h \in 2^{\mathcal{C}} \quad n \in \mathbb{N}}{(h, n) \in \mathcal{C}} \quad \frac{\text{(ELEM)} \quad x \in \mathcal{T} \cup \mathcal{P} \quad h \in 2^{\mathcal{C}}}{x(h) \in \mathcal{C}}$$

Colours resemble the unfolding construction (Figure 3): the colours for tokens are (h, n) , where h denotes its (possible empty) set of causes and n is a natural number used for distinguishing tokens with identical causal history. Causal histories are build from coloured versions of transitions (i.e., $t(h)$) and places (i.e., $a(h)$).

Definition 11 (P/T as C-P/T). Let $N = (S_N, T_N, \bullet_{-N}, \bullet_N)$ be a P/T net. Then, $\llbracket N \rrbracket$ is the C-P/T defined such that $\llbracket N \rrbracket = (S_N, T_N, \bullet_{-\llbracket N \rrbracket}, \bullet_{\llbracket N \rrbracket})$ and

- $\bullet_{\llbracket N \rrbracket} = a_1(x_1) \oplus \dots \oplus a_n(x_n)$ where $\bullet_{t_N} = a_1 \dots a_n$ and $\forall 1 \leq i \leq n. x_i \in \mathcal{X}$.
- $\bullet_{\llbracket N \rrbracket} = \{a(\{t(h)\}, i) \mid a \in \text{supp}(t_N^\bullet) \wedge 1 \leq i \leq t_N^\bullet(a) \wedge h = \bullet_{\llbracket N \rrbracket}\}$.

A marked net (N, m) is encoded as $\llbracket (N, m) \rrbracket = (\llbracket N \rrbracket, \llbracket m \rrbracket)$ where $\llbracket m \rrbracket = \{a(\emptyset, i) \mid a \in \text{supp}(m) \wedge 1 \leq i \leq m(a)\}$.

The encoding does not alter the structure of a net; it only adds colours to its tokens. In fact, an encoded net has the same places and transitions as the original net, and pre- and postsets of each transition have the same support. Added colours do not interfere with firing because the preset of each transition uses different colour variables for different tokens. The colour $\{t(h)\}$ assigned to each token produced by the firing of t describes the causal history of the token, i.e., it indicates that the token has been produced by t after consuming the tokens in the preset of t , which is denoted by h . The natural number i is used for distinguishing multiple tokens produced by the same firing. Tokens in the initial marking are coloured as (\emptyset, i) , i.e., they have empty causal history.

Example 5. The encoding of the nets in Figure 2 are shown in Figure 6. We comment on the encoding of N_1 . The transition $t_1 = a \rhd c$ in N_1 is encoded as $a(x) \rhd c(t_1(a(x)), 1)$, i.e., the firing of t_1 that consumes a token with colour h from place a generates a token in c with colour $(t_1(a(h)), 1)$. The transition $t_2 = b \oplus c \rhd e$ has two places in the preset and uses two variables x and y in its encoded form $b(x) \oplus c(y) \rhd e(t_2(b(x) \oplus c(y)), 1)$. Note that the colour of the token produced in c carries the information of the tokens consumed from both places b and c . The encoding for t_3 is defined analogously.

We illustrate a sequence of firings of $\llbracket (N_1, \mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c} \oplus \mathbf{d}) \rrbracket$.

$$\begin{aligned} & (\llbracket N_1 \rrbracket, \mathbf{a}(\emptyset, 1) \oplus \mathbf{b}(\emptyset, 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{d}(\emptyset, 1)) \\ & \xrightarrow{t_1} (\llbracket N_1 \rrbracket, \mathbf{b}(\emptyset, 1) \oplus \mathbf{c}(t_1(\mathbf{a}(\emptyset, 1)), 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{d}(\emptyset, 1)) \\ & \xrightarrow{t_2} (\llbracket N_1 \rrbracket, \mathbf{e}(t_2(\mathbf{b}(\emptyset, 1) \oplus \mathbf{c}(t_1(\mathbf{a}(\emptyset, 1)), 1)), 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{d}(\emptyset, 1)) \end{aligned}$$

The firing of t_1 consumes the token $(\emptyset, 1)$ from \mathbf{a} and produces the token $(t_1(\mathbf{a}(\emptyset, 1)), 1)$ in place \mathbf{c} . The causal history of the token $t_1(\mathbf{a}(\emptyset, 1))$ indicates that the token has been produced by the firing of t_1 that consumed the token $(\emptyset, 1)$ from \mathbf{a} . The second reduction takes place because of the firing of t_2 . By inspecting the causal history of the token produced in the place \mathbf{e} we can conclude that t_2 has consumed the token previously generated by t_1 .

The following result shows that there is a tight correspondence between the semantics of the coloured version of a P/T net and its unfolding.

Lemma 8. *Let (N, m) be a marked P/T net and $\mathcal{U}[N, m] = (O, m')$ its unfolding. Then, $\llbracket N, m \rrbracket \xrightarrow{s} (\llbracket N \rrbracket, m'')$ iff $(O, m') \xrightarrow{s} (O, m'')$.*

Proof. The *if* part follows by induction on the length of the reduction. The base case follows by taking $m'' = m'$ and noting that $\llbracket N, m \rrbracket = (\llbracket N \rrbracket, m')$. The inductive step $s = s'; t$ follows by applying inductive hypothesis on s' to conclude that $\llbracket N, m \rrbracket \xrightarrow{s'} (\llbracket N \rrbracket, m''')$ iff $(O, m) \xrightarrow{s'} (O, m''')$. If $(\llbracket N \rrbracket, m''') \xrightarrow{t} (\llbracket N \rrbracket, m'')$ implies $m''' = \bullet t_{\llbracket N \rrbracket} \oplus m''''$ and $m'' = t_{\llbracket N \rrbracket} \bullet \oplus m''''$. Since $(O, m) \xrightarrow{s'} (O, m''')$, $CO(\bullet t)$. Then, by the unfolding construction we conclude $(O, m''') \xrightarrow{t} (O, m'')$. The *only if* follows analogously.

The reversible version of $\llbracket N \rrbracket$ is defined as for occurrence nets, by adding transitions that are the swapped versions of the ones in N .

Definition 12 (Reversible P/T net). *Let N be a P/T net. The reversible version of N is $\overleftarrow{\llbracket N \rrbracket}$. The reversible version of a marked P/T net (N, m) is the marked C-P/T net $(\overleftarrow{\llbracket N \rrbracket}, \llbracket m \rrbracket)$.*

Example 6. The net $\overleftarrow{\llbracket N_2 \rrbracket}$, the reversible version of $\llbracket N_2 \rrbracket$ from Figure 6, is shown in Figure 7. We now illustrate the execution of $\overleftarrow{\llbracket N_2 \rrbracket}$.

$$\begin{aligned} & (\overleftarrow{\llbracket N_2 \rrbracket}, \mathbf{a}(\emptyset, 1) \oplus \mathbf{b}(\emptyset, 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{d}(\emptyset, 1)) \\ & \xrightarrow{t_1} (\overleftarrow{\llbracket N_2 \rrbracket}, \mathbf{b}(\emptyset, 1) \oplus \mathbf{b}(t_1(\mathbf{a}(\emptyset, 1)), 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{d}(\emptyset, 1)) \\ & \xrightarrow{t_2} (\overleftarrow{\llbracket N_2 \rrbracket}, \mathbf{b}(t_1(\mathbf{a}(\emptyset, 1)), 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{e}(\mathbf{b}(\emptyset, 1)\mathbf{c}(\emptyset, 1), 1) \oplus \mathbf{d}(\emptyset, 1)) \\ & \xrightarrow{\overleftarrow{t_1}} (\overleftarrow{\llbracket N_2 \rrbracket}, \mathbf{a}(\emptyset, 1) \oplus \mathbf{e}(\mathbf{b}(\emptyset, 1)\mathbf{c}(\emptyset, 1), 1) \oplus \mathbf{d}(\emptyset, 1)) \\ & \xrightarrow{\overleftarrow{t_2}} (\overleftarrow{\llbracket N_2 \rrbracket}, \mathbf{a}(\emptyset, 1) \oplus \mathbf{b}(\emptyset, 1) \oplus \mathbf{c}(\emptyset, 1) \oplus \mathbf{d}(\emptyset, 1)) \end{aligned}$$

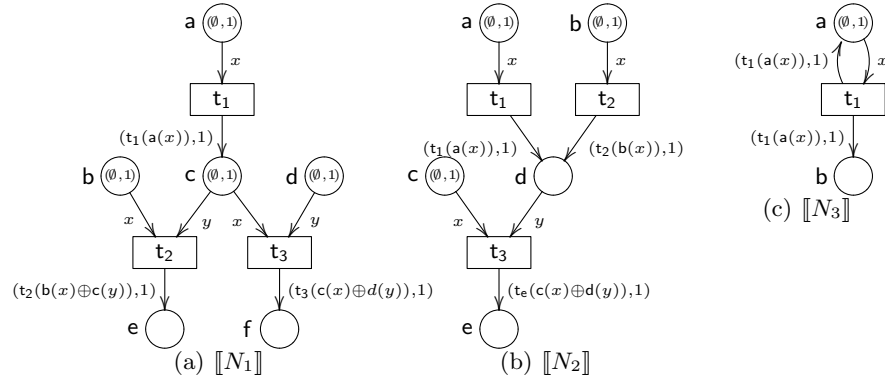
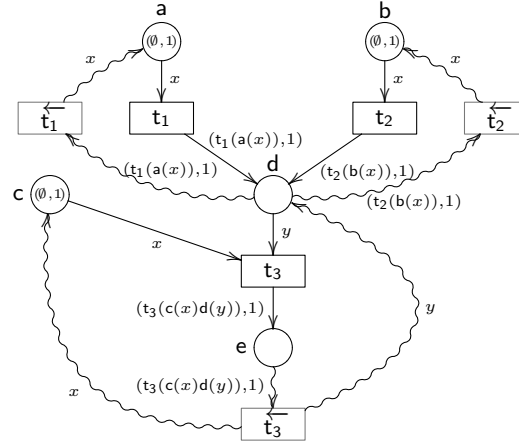


Fig. 6. P/T nets as C-P/T nets

Fig. 7. Reversible coloured net $\overleftarrow{N_2}$.

In the example above, the firing t_2 can choose to consume either the token $b(\emptyset, 1)$ or the token $b(t_1(a(\emptyset, 1)), 1)$. Since the first one is chosen, then after t_2 it is still possible to undo t_1 . If t_2 chose the second token, then in order to undo t_1 we would first undo t_2 , since firing t_1 is not enabled by the token $a(\emptyset, 1)$.

The following result states that the reductions of the reversible C-P/T of a net are in one-to-one correspondence with the reductions of its reversible unfolding.

Theorem 3 (Correctness). *Let (N, m) be a marked P/T net and $\mathcal{U}[N, m] = (O, m')$ its unfolding. Then, $\overleftarrow{N}, [m] \xrightarrow{s} ([N], m'')$ iff $(\overleftarrow{O}, m') \xrightarrow{s} (\overleftarrow{O}, m'')$.*

Proof. It follows by induction on the length of the reduction analogously to the proof of Lemma 8.

7 Conclusions

We have presented a causally reversible semantics for Place/Transitions Petri Nets (P/T nets) based on two observations. First, occurrence net can be straightforwardly reversed by adding for each transition its reverse. Second, the standard unfolding construction associates a P/T net with an occurrence net that preserves all of its computation. Consequently, the reversible semantics of a P/T net can be obtained as the reversible semantics of its unfolding. We have showed that reversibility in reversible occurrence net is causal-consistent, that is it preserves causality. The unfolding of an occurrence net can be infinite (e.g., if the original P/T net is not acyclic). Therefore we have shown that the reversible behaviour of reversible occurrence nets can be expressed as a finite net whose tokens are coloured by causal histories. Colours in our encoding resemble the causal memories that are typical in reversible process calculi [12,6].

Occurrence nets have a direct mapping into prime event structures. We shall investigate in the future the relation between reversible event structures [22,5,26,8] and our reversible occurrence nets. There is an alternative method for proving causally-consistent reversibility in a reversible model of computation. It is based on showing other properties than those in Section 4, mainly the well-foundedness (lack of infinite reverse sequences) and Reverse Diamond properties [21,20]. It would be worthwhile to prove the alternative properties for our reversible nets, and compare the two approaches.

References

1. Kamila Barylska, Anna Gogolinska, Lukasz Mikulski, Anna Philippou, Marcin Piatkowski, and Kyriaki Psara. Reversing computations modelled by coloured petri nets. In Wil M. P. van der Aalst, Robin Bergenthum, and Josep Carmona, editors, *Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data*, volume 2115 of *CEUR Workshop Proceedings*, pages 91–111. CEUR-WS.org, 2018.
2. Kamila Barylska, Maciej Koutny, Lukasz Mikulski, and Marcin Piatkowski. Reversible computation vs. reversibility in petri nets. *Sci. Comput. Program.*, 151:48–60, 2018.
3. E. Cardoza, R. Lipton, and A. R. Meyer. Exponential space complete problems for petri nets and commutative semigroups (preliminary report). In *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, STOC '76, pages 50–54, New York, NY, USA, 1976. ACM.
4. Ioana Cristescu, Jean Krivine, and Daniele Varacca. A compositional semantics for the reversible p-calculus. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, pages 388–397. IEEE Computer Society, 2013.
5. Ioana Domnina Cristescu, Jean Krivine, and Daniele Varacca. Rigid families for CCS and the π -calculus. In Martin Leucker, Camilo Rueda, and Frank D. Valencia, editors, *Theoretical Aspects of Computing - ICTAC 2015 - 12th International Colloquium*, volume 9399 of *Lecture Notes in Computer Science*, pages 223–240. Springer, 2015.
6. Vincent Danos and Jean Krivine. Reversible communicating systems. In Philippa Gardner and Nobuko Yoshida, editors, *CONCUR 2004 - Concurrency Theory, 15th International Conference*, volume 3170 of *Lecture Notes in Computer Science*, pages 292–307. Springer, 2004.
7. Elena Giachino, Ivan Lanese, and Claudio Antares Mezzina. Causal-consistent reversible debugging. In Stefania Gnesi and Arend Rensink, editors, *Fundamental Approaches to Software Engineering - 17th International Conference, FASE 2014*, volume 8411 of *Lecture Notes in Computer Science*, pages 370–384. Springer, 2014.
8. Eva Graversen, Iain Phillips, and Nobuko Yoshida. Event structure semantics of (controlled) reversible CCS. In Jarkko Kari and Irek Ulidowski, editors, *Reversible Computation - 10th International Conference*, volume 11106 of *Lecture Notes in Computer Science*, pages 102–122. Springer, 2018.
9. Jonathan Hayman and Glynn Winskel. The unfolding of general petri nets. In Ramesh Hariharan, Madhavan Mukund, and V. Vinay, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2008, December 9-11, 2008, Bangalore, India*, volume 2 of *LIPIcs*, pages 223–234. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2008.
10. James Hoey, Irek Ulidowski, and Shoji Yuen. Reversing imperative parallel programs with blocks and procedures. In *Proceedings of Express/SOS*, 2018, 2018.
11. Stefan Kuhn and Irek Ulidowski. A calculus for local reversibility. In Simon J. Devitt and Ivan Lanese, editors, *Reversible Computation - 8th International Conference, RC 2016*, volume 9720 of *Lecture Notes in Computer Science*, pages 20–35. Springer, 2016.
12. Ivan Lanese, Claudio Antares Mezzina, and Jean-Bernard Stefani. Reversibility in the higher-order π -calculus. *Theor. Comput. Sci.*, 625:25–84, 2016.
13. Ivan Lanese, Claudio Antares Mezzina, and Francesco Tiezzi. Causal-consistent reversibility. *Bulletin of the EATCS*, 114, 2014.

14. Ivan Lanese, Naoki Nishida, Adrián Palacios, and Germán Vidal. Cauder: A causal-consistent reversible debugger for Erlang. In John P. Gallagher and Martin Sulzmann, editors, *Functional and Logic Programming - 14th International Symposium, FLOPS*, volume 10818 of *Lecture Notes in Computer Science*, pages 247–263. Springer, 2018.
15. George B. Leeman, Jr. A formal approach to undo operations in programming languages. *ACM Trans. Program. Lang. Syst.*, 8(1):50–87, January 1986.
16. Jean-Jacques Lévy. An algebraic interpretation of the λ -calculus; and an application of a labelled λ -calculus. *Theor. Comput. Sci.*, 2(1):97–114, 1976.
17. Doriana Medic, Claudio Antares Mezzina, Iain Phillips, and Nobuko Yoshida. A parametric framework for reversible pi-calculi. In Jorge A. Pérez and Simone Tini, editors, *Proceedings Combined 25th International Workshop on Expressiveness in Concurrency and 15th Workshop on Structural Operational Semantics and 15th Workshop on Structural Operational Semantics, EXPRESS/SOS*, volume 276 of *EPTCS*, pages 87–103, 2018.
18. Mogens Nielsen, Gordon D. Plotkin, and Glynn Winskel. Petri nets, event structures and domains, part I. *Theor. Comput. Sci.*, 13:85–108, 1981.
19. Anna Philippou and Kyriaki Psara. Reversible computation in petri nets. In Jarkko Kari and Irek Ulidowski, editors, *Reversible Computation - 10th International Conference, RC 2018*, volume 11106 of *Lecture Notes in Computer Science*, pages 84–101. Springer, 2018.
20. Iain Phillips and Irek Ulidowski. Reversibility and models for concurrency. In *Proceedings of 4th Workshop on Structural Operational Semantics SOS 2007*, volume 192 of *ENTCS*, pages 93–108, 2007.
21. Iain Phillips and Irek Ulidowski. Reversing algebraic process calculi. *J. Log. Algebr. Program.*, 73(1-2):70–96, 2007.
22. Iain Phillips and Irek Ulidowski. Reversibility and asymmetric conflict in event structures. *J. Log. Algebr. Meth. Program.*, 84(6):781–805, 2015.
23. Iain Phillips, Irek Ulidowski, and Shoji Yuen. A reversible process calculus and the modelling of the ERK signalling pathway. In *Proceedings of Reversible Computation 2012*, volume 7581 of *LNCSS*, pages 218–232. Springer, 2013.
24. G. Michele Pinna. Reversing steps in membrane systems computations. In Marian Gheorghe, Grzegorz Rozenberg, Arto Salomaa, and Claudio Zandron, editors, *Membrane Computing - 18th International Conference, CMC 2017*, volume 10725 of *Lecture Notes in Computer Science*, pages 245–261. Springer, 2017.
25. Markus Schordan, Tomas Oppelstrup, David R. Jefferson, and Peter D. Barnes Jr. Generation of reversible C++ code for optimistic parallel discrete event simulation. *New Generation Comput.*, 36(3):257–280, 2018.
26. Irek Ulidowski, Iain Phillips, and Shoji Yuen. Reversing event structures. *New Generation Comput.*, 36(3):281–306, 2018.
27. Alexis De Vos, Stijn De Baerdemacker, and Yvan Van Rentergem. *Synthesis of Quantum Circuits vs. Synthesis of Classical Reversible Circuits*. Synthesis Lectures on Digital Circuits and Systems. Morgan & Claypool Publishers, 2018.

A Proofs

We start with the proof of a simple version of Square Lemma from Section 2. Note that t and t' are forward transitions.

Lemma 1 *Let t and t' be coinitial concurrent transitions. Then, there exist transitions t_1 and t'_1 such that $t; t'_1$ and $t'; t_1$ are cofinal.*

Proof. Since t and t' be coinitial they do not cause each other. To be concurrent they also cannot be in an immediate conflict, implying that we can write them as $(N, m \oplus m' \oplus m'') \xrightarrow{t} (N, m_1 \oplus m' \oplus m'')$ and $(N, m \oplus m' \oplus m'') \xrightarrow{t'} (N, m \oplus m'_1 \oplus m'')$, where m and m' do not overlap. Clearly then these are valid transitions: $(N, m \oplus m'_1 \oplus m'') \xrightarrow{t_1} (N, m_1 \oplus m'_1 \oplus m'')$ and $(N, m_1 \oplus m' \oplus m'') \xrightarrow{t'_1} (N, m_1 \oplus m'_1 \oplus m'')$. Since t_1 and t'_1 are cofinal (resulting in $(N, m_1 \oplus m'_1 \oplus m'')$), we obtain $t; t'_1$ and $t'; t_1$ are cofinal.

We now report the proofs of the main Lemmas/Theorems of Section 4.

First, we have a lemma which is instrumental for the proof of causal-consistent reversibility in reversible calculi [6,12]. Here, each of t and t' is either a forward or reverse transition.

Lemma 4 *Let t and t' be coinitial concurrent transitions. Then, there exist transitions t_1 and t'_1 such that $t; t'_1$ and $t'; t_1$ are cofinal.*

Proof. We consider three cases when t and t' are coinitial concurrent transitions. If t and t' are both forward then we are done by lemma 1. If t and t' are both reverse, then since they are coinitial they cannot cause one another. They cannot be in an immediate conflict because this would imply a backward conflict on their forward versions (which we do not have in occurrence nets). So, t, t' do not share tokens in their preplaces. Hence, we can write t, t' correspondingly as we wrote t, t' in the proof of lemma 1. The rest of the case follows correspondingly as in the proof of lemma 1. If t is a transition and t' is a reverse transition, then since they are coinitial they cannot cause one another. Hence, for t and t' to be concurrent we require $\overleftarrow{t'} \not\leq t$. This means that the preplaces of t do not overlap with the postplaces of $\overleftarrow{t'}$, hence the preplaces of t do not overlap with the preplaces of t' . We then finish as in the proof of lemma 1.

Lemma 5 *Let ω be a trace. There exist two forward traces ω_1 and ω_2 such that $\omega \asymp \overleftarrow{\omega_1}; \omega_2$.*

Proof. The proof is by lexicographic induction on the length of ω and on the distance between the beginning of ω and the *earliest pair* of firings in ω of the form $t_1; \overleftarrow{t_2}$, where t_1 and t_2 are forward. If such pair does not exist, then the result follows immediately. If there exists one such pair we have two possibilities: either the two firings are concurrent or they are not.

In the first case, this implies that $\bullet t_1 \cap \bullet \overleftarrow{t_2} = \emptyset$. By exploiting the property of Mazurkiewicz traces we can swap the two firings, obtaining a later earliest

pair contradicting our assumption. The result then follows by induction since swapping firings keeps the total length of the trace unchanged.

If the two firings are not concurrent we have four cases:

1. $t_1 \prec \overleftarrow{t_2}$, this implies that $t_1 \bullet \cap \bullet \overleftarrow{t_2} \neq \emptyset$ and that $t_1 \bullet \cap t_2 \bullet \neq \emptyset$. This means that there is a backward conflict between t_1 and t_2 . By definition occurrence nets are backward conflict free: contradiction.
2. $\overleftarrow{t_2} \prec t_1$, this implies that $\overleftarrow{t_2} \bullet \cap \bullet t_1 \neq \emptyset$, and that $\bullet t_2 \cap \bullet t_1 \neq \emptyset$. The last says that t_1 and t_2 are in an immediate conflict. Since we have $t_1; \overleftarrow{t_2}$ in ω it means that in order to undo t_2 the transition must have occurred before t_1 in ω . It is not possible that some other transition t_3 produced a token to a place a that was used by $\overleftarrow{t_2}$: if it could happen it would mean that $\{t_2, t_3\} = \bullet a$, so there is a backward conflict: contradiction. Hence, t_1 and t_2 occurred before $\overleftarrow{t_2}$ in ω . Since t_1 and t_2 are in an immediate conflict, if one transition takes place the other cannot happen (as they share tokens in their preplaces). This contradicts that both t_1 and t_2 appear in ω before $\overleftarrow{t_2}$.
3. $t_1 \#_0 \overleftarrow{t_2}$ and by definition of $\#_0$ we have $\bullet t_1 \cap \bullet \overleftarrow{t_2}$. Since $t_1; \overleftarrow{t_2}$ appears in ω , the firing t_1 uses some tokens needed by $\overleftarrow{t_2}$ to fire. Since there is no computation between t_1 and $\overleftarrow{t_2}$ in ω (that can replace the tokens needed by $\overleftarrow{t_2}$), $\overleftarrow{t_2}$ cannot fire: contradiction.
4. $t_1 \# \overleftarrow{t_2}$, this implies that there exist earlier x and y in ω such that $x \prec t_1$ and $y \prec \overleftarrow{t_2}$ with $x \#_0 y$. So we have x and y in an immediate conflict appearing in ω prior to subtrace $t_1; \overleftarrow{t_2}$. We then show a contradiction as in case 2 above.

□

Lemma 6 *Let $\omega_1 \asymp \omega_2$ with ω_2 forward. Then, $|\omega_2| \leq |\omega_1|$.*

Proof. We prove this lemma by induction on the length of ω_1 . If ω_1 is a forward trace we are already done. Otherwise by applying Lemma 5 we have that $\omega_1 \asymp \overleftarrow{\omega}; \omega'$ (with both ω, ω' forward). Consider the sub-trace $\overleftarrow{t_1}; t_2$ with two successive firings with opposite directions in $\overleftarrow{\omega}; \omega'$. Clearly it is the only sub-trace of this form in $\overleftarrow{\omega}; \omega'$. Suppose $\bullet \overleftarrow{t_1} = m'$ and $\overleftarrow{t_1} \bullet = m''$. Since ω_2 is a forward only trace, this means that the marking m' removed by $\overleftarrow{t_1}$ have to be put back in ω_1 , otherwise this change would be visible in ω_2 (since it is a forward only trace). Let t_3 the earliest transition able to consume m'' and put back m' . This implies that $t_3 = t_1$ since in occurrence nets there are no loops. Let us note that we can apply this reasoning to occurrence nets since we just consider forward firings. Moreover, all the transitions between $\overleftarrow{t_1}$ and t_3 can be swapped with t_3 , since they are concurrent with respect to t_3 . This is because all of them are forward transitions, occurrence nets are free from loops, and since t_3 is the first firing after $\overleftarrow{t_1}$ able to consume m'' and produce m' . We can then apply the property of Mazurkiewicz traces, and obtain an equivalent trace where $\overleftarrow{t_1}$ and t_3 are adjacent. Next, by applying \asymp , we can cancel them out (since $t_3 = t_1$). Finally we conclude by applying the inductive hypothesis on a shorter trace. □

Theorem 1

$$\omega_1 \asymp \omega_2 \text{ if and only if } (\overleftarrow{\mathcal{O}}, m_0) \xrightarrow{\omega_1} (\overleftarrow{\mathcal{O}}, m_n) \iff (\overleftarrow{\mathcal{O}}, m_0) \xrightarrow{\omega_2} (\overleftarrow{\mathcal{O}}, m_n)$$

Proof. \Rightarrow **direction.** If $\omega_1 \asymp \omega_2$ then $(N, m_0) \xrightarrow{\omega_1} (\overleftarrow{O}, m_n) \iff (\overleftarrow{O}, m_0) \xrightarrow{\omega_2} (\overleftarrow{O}, m_n)$. First, notice that if $\omega_1 \asymp \omega_2$ then it must be the case that ω_1 can be transformed into ω_2 (and vice versa) through $n \geq 0$ applications of the rules of \asymp . We then proceed by induction on n . In the base case, $n = 0$, we have that $\omega_1 \asymp \omega_2$ by applying 0 times the rules of \asymp . Since \asymp is an equivalence, this means that $\omega_1 = \omega_2$ which in turn implies that the traces are coinital and cofinal. In the inductive case, we have that there exist n traces ω^k (with $0 \leq k \leq n$) obtained as a result of applying the rules of \asymp to ω_1 exactly k times; hence, $\omega^0 = \omega_1$ and $\omega^n = \omega_2$. We then have that $\omega^{n-1} \asymp \omega_2$, i.e., traces ω^{n-1} and ω_2 differ in one axiom application; this means that we can decompose both traces as follows:

$$\omega^{n-1} = \omega_a; \omega'; \omega_b \quad \omega_2 = \omega_a; \omega''; \omega_b$$

with ω' and ω'' differing by just one application of \asymp . We have then three cases:

1. $\omega' = \mathbf{t}_1; \mathbf{t}_2$ and $\omega'' = \mathbf{t}_2; \mathbf{t}_1$ with \mathbf{t}_1 co \mathbf{t}_2
2. $\omega' = \mathbf{t}_i; \mathbf{t}$ and $\omega'' = \epsilon_m$ with m the marking before firing \mathbf{t}
3. $\omega' = \mathbf{t}; \mathbf{t}$ and $\omega'' = \epsilon_m$ with m the marking after firing \mathbf{t}

In all cases it is easy to see that ω^{n-1} and ω_2 are both coinital and cofinal. By inductive hypothesis, $\omega_1 \asymp \omega^{n-1}$ implies that ω_1 and ω^{n-1} are coinital and cofinal; since $\omega^{n-1} \asymp \omega_2$ implies that they are coinital and cofinal, we can conclude that also ω_1 and ω_2 are coinital and cofinal if $\omega_1 \asymp \omega_2$.

\Leftarrow **direction.** We have to show that if $(\overleftarrow{O}, m_0) \xrightarrow{\omega_1} (\overleftarrow{O}, m_n)$ and $(\overleftarrow{O}, m_0) \xrightarrow{\omega_2} (\overleftarrow{O}, m_n)$ then $\omega_1 \asymp \omega_2$. By Lemma 5 we know that the two traces can be written as composition of a backward trace and a forward one. The proof is by lexicographic induction on the sum of the lengths of ω_1 and ω_2 and on the distance between the end of ω_1 and the earliest pair of transitions \mathbf{t}_1 in ω_1 and \mathbf{t}_2 in ω_2 which are not equal. If all the transitions are equal then we are done. Otherwise we have to consider four cases depending on the direction of the two transitions.

t₁ forward and t₂ backward : we have that $\omega_1 = \overleftarrow{\omega}; \mathbf{t}_1; \omega'$ and $\omega_2 = \overleftarrow{\omega}; \mathbf{t}_2; \omega''$ with $\overleftarrow{\omega}$ the common backward subtrace and $\mathbf{t}_1; \omega'$ a forward trace. By hypothesis we have that ω_1 and ω_2 are coinital and cofinal and this implies that also $\mathbf{t}_1; \omega'$ and $\mathbf{t}_2; \omega''$ are coinital and cofinal. By applying Lemma 6 on the trace $\mathbf{t}_2; \omega''$ we obtain a shorter equivalent forward trace and therefore ω_2 has a shorter causally equivalent forward trace. We can then conclude by induction (on a shorter trace).

t₁ backward and t₂ forward : similar to the previous case.

t₁ forward and t₂ forward By assumption the two transitions are different, that $\mathbf{t}_1 \neq \mathbf{t}_2$. We can decompose the two traces as follows $\omega_1 = \omega; \mathbf{t}_1; \omega^1$ and $\omega_2 = \omega; \mathbf{t}_2; \omega^2$. Moreover all the firings in ω^i are forward. If \mathbf{t}_1 and \mathbf{t}_2 are in conflict, this means that $\bullet \mathbf{t}_1 \cap \bullet \mathbf{t}_2 \neq \emptyset$. Since the two traces are cofinal, this means that the effect of \mathbf{t}_1 will remain visible till the end, so we have:

$$\begin{aligned} (\overleftarrow{O}, m_0) &\xrightarrow{\omega} (\overleftarrow{O}, m) \xrightarrow{\mathbf{t}_1} (\overleftarrow{O}, m_1) \xrightarrow{\omega^1} (\overleftarrow{O}, m_f) \\ (\overleftarrow{O}, m_0) &\xrightarrow{\omega} (\overleftarrow{O}, m) \xrightarrow{\mathbf{t}_2} (\overleftarrow{O}, m_2) \xrightarrow{\omega^2} (\overleftarrow{O}, m_f) \end{aligned}$$

with $m_1 \neq m_2$. But this implies that in ω^2 there will be a firing able to put produce the tokens necessary to t_1 to fire, which implies the existence of a loop. Since we are just considering the forward firing of a reversible occurrence net, they are by construction the transition of an occurrence net, which by definition is acyclic. So this case cannot happen. Hence we have that $t_1 \text{ co } t_2$, and that $t_1 \in \omega^2$. We now have to show that t_1 is concurrent with all the firings between t_2 and itself. We can apply the same reasoning as before (absence of cycles) and by applying several times the property of Mazurkiewicz traces we can rearrange ω_2 in such a way that $\omega_2 = \omega; t_1; t_2; \omega^3$, where ω^3 is ω^2 without t_2 . We then have $\omega_1 = \omega; t_1; \omega^1$ and $\omega_2 = \omega; t_1; t_2; \omega^3$. They have the same length as before, but the first pair of discording firings is closer to the end of the trace. The thesis follows by inductive hypothesis.

t_1 backward and t_2 backward the two firings cannot remove the same token, since this would mean that by considering their forward variants there is a backward conflict, which by definition occurrence net is impossible. Since the two traces are coinital and cofinal, then either there exists in ω_1 a firing which is the forward version of t_1 or there exists in ω_2 the firing t_1 . In the first case, t_1 is concurrent with all the backward transitions following it (in the trace) but with the ones which do the same firing. All these transitions have a corresponding forward move in the trace, otherwise we would have that the effect of t_1 is visible till the end, contradicting the hypothesis that ω_1 and ω_2 are coinital and cofinal. We then select the last of such firing and swap it with all the backward firings following it, in order to make it the very last backward firing of the trace. We then select the first forward firing which is its opposite, and since it is concurrent with all the forward firings proceeding it, we can swap them in such a way to have it as the very first forward firing. We can then apply the axiom $\overleftarrow{t_1}; t_1 \asymp \epsilon_m$ with m being the marking of the net before firing $\overleftarrow{t_1}$. We then obtain a shorter trace, equivalent to ω_1 . The thesis then follows by induction.

In the second case, the reasoning is similar to the case in which both t_1 and t_2 are forward.

Theorem 2 2] *Let O be an occurrence net and m_0 an initial marking. Then,*

$$(\overleftarrow{O}, m_0) \rightarrow^* (\overleftarrow{O}, m') \iff (\overleftarrow{O}, m_0) \twoheadrightarrow^* (\overleftarrow{O}, m')$$

Proof. The \Leftarrow case trivially holds, as $\rightarrow \subset \twoheadrightarrow$. In the other case we have that $(\overleftarrow{O}, m_0) \twoheadrightarrow (\overleftarrow{O}, m')$. By using Lemma 5 we have that $\omega \asymp \overleftarrow{\omega_1}; \omega_2$ with ω_2 forward. Since $\omega \asymp \overleftarrow{\omega_1}; \omega_2$, by Theorem 1 we have that also $(\overleftarrow{O}, m_0) \xrightarrow{\omega_1 \omega_2} (\overleftarrow{O}, m')$. Let us note that m_0 is an initial marking, and this implies that no backward computations can take place from (\overleftarrow{O}, m_0) meaning that $\omega_1 = \epsilon_{m_0}$. Hence we have that $\omega \asymp \omega_2$ with ω_2 a forward trace. By Theorem 1 we have that $(O, m_0) \xrightarrow{\omega} (O, m')$ implies $(O, m_0) \xrightarrow{\omega_2} (O, m')$ as desired. \square

B Additional Figures

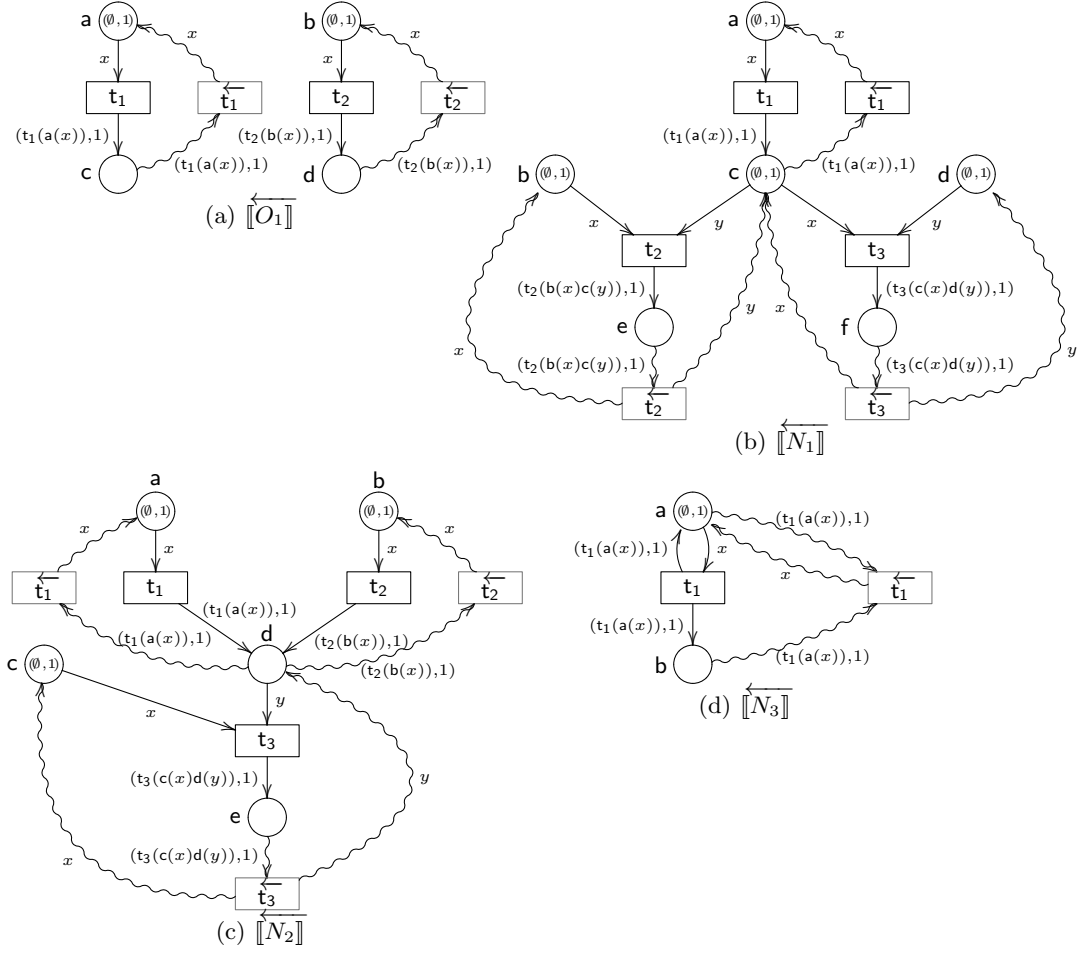


Fig. 8. Reversible coloured nets