Lending Petri Nets and Contracts

Massimo Bartoletti, Tiziana Cimoli, G. Pinna

Lending Petri nets and contracts

Massimo Bartoletti, Tiziana Cimoli, and G. Michele Pinna

Dipartimento di Matematica e Informatica, Università degli Studi di Cagliari, Italy

Abstract. Choreography-based approaches to service composition typically assume that, after a set of services has been found which correctly play the roles prescribed by the choreography, each service respects his role. Honest services are not protected against adversaries. We propose a model for contracts based on an extension of Petri nets, which allows services to protect themselves while still realizing the choreography. We relate this model with Propositional Contract Logic, by showing a translation of formulae into our Petri nets which preserves the logical notion of agreement, and allows for compositional verification.

1 Introduction

Many of today's human activities, from business and financial transactions, to collaborative and social applications, run over complex interorganizational systems, based on service-oriented computing (SOC) and cloud computing technologies. These technologies foster the implementation of complex software systems through the composition of basic building blocks, called *services*. Ensuring reliable coordination of such components is fundamental to avoid critical, possibly irreparable problems, ranging from economic losses in case of commercial activities, to risks for human life in case of safety-critical applications.

Ideally, in the SOC paradigm an application is constructed by dynamically discovering and composing services published by different organizations. Services have to *cooperate* to achieve the overall goals, while at the same time they have to *compete* to achieve the specific goals of their stakeholders. These goals may be conflicting, especially in case of mutually distrusted organizations. Thus, services must play a double role: while cooperating together, they have to protect themselves against other service's misbehavior (either unintentional or malicious).

The lack of precise guarantees about the reliability and security of services is a main deterrent for industries wishing to move their applications and business to the cloud [3]. Quoting from [3], "absent radical improvements in security technology, we expect that users will use contracts and courts, rather than clever security engineering, to guard against provider malfeasance".

Indeed, contracts are already a key ingredient in the design of SOC applications. A *choreography* is a specification of the overall behavior of an interorganizational process. This *global* view of the behavior is projected into a set of *local* views, which specify the behavior expected from each service involved in the whole process. The local views can be interpreted as the service contracts: if

the actual implementation of each service respects its contract, then the overall application must be guaranteed to behave correctly.

There are many proposals of formal models for contracts in the literature, which we may roughly divide into "physical" and "logical" models. Physical contracts take inspiration mainly from formalisms for concurrent systems (e.g. Petri nets [21], event structures [15, 5], and various sorts of process algebras [8–10, 12, 16]), and they allow to describe the interaction of services in terms of response to events, message exchanges, *etc.* On the other side, logical contracts are typically expressed as formulae of suitable logics, which take inspiration and extend e.g. modal [1, 14], intuitionistic [2, 7], linear [2], deontic [18] logics to model high-level concepts such as promises, obligations, prohibitions, authorizations, *etc.*

Even though logical contracts are appealing, since they aim to provide formal models and reasoning tools for real-world Service Level Agreements, existing logical approaches have not had a great impact on the design of SOC applications. A reason is that there is no evidence on how to relate high-level properties of a contract with properties of the services which have to realize it. The situation is decidedly better in the realm of physical contracts, where the gap between contracts and services is narrower. Several papers, e.g. [9–11, 16, 21], address the issue of relating properties of a choreography with properties of the services which implement it (e.g. deadlock freedom, communication error freedom, session fidelity), in some cases providing automatic tools to project the choreography to a set services which correctly implements it.

A common assumption of most of these approaches is that services are *honest*, i.e. their behavior always adheres to the local view. For instance, if the local view takes the form of a behavioral type, it is assumed that the service is typeable, and that its type is a subtype of the local view. Contracts are only used in the "matchmaking" phase: once, for each local view projected from the choreography, a compliant service has been found, then all the contracts can be discarded.

We argue that the honesty assumption is not suitable in the case of interorganizational processes, where services may pursue their providers goals to the detriment to the other ones. For instance, consider a choreography which prescribes that a participant A performs action a (modeling e.g. "pay \$100 to B"), and that B performs b (e.g. "provide A with 5GB disk storage"). If both A and B are honest, then each one will perform its due action, so leading to a correct execution of the choreography. However, since providers have full control of the services they run, there is no authority which can force services to be honest. So, a malicious provider can replace a service validated w.r.t. its contract, with another one: e.g., B could wait until A has done a, and then "forget" to do b. Note that B may perform his scam while not being liable for a contract violation, since contracts have been discarded after validation.

In such competitive scenarios, the role of contracts is twofold. On the one hand, they must guarantee that their composition complies with the choreography: hence, in contexts where services are honest, the overall execution is correct. On the other hand, contracts must protect services from malicious ones: in the

example above, the contract of A must ensure that, if A performs a, then B will either do b, or he will be considered culpable of a contract violation.

In this paper, we consider physical contracts modeled as Petri nets, along the lines of [21]. In our approach we can both start from a choreography (modeled as a Petri net) and then obtain the local views by projection, as in [21], or start from the local views, i.e. the contracts published by each participant, to construct a choreography which satisfies the goals of everybody. Intuitively, when this happens the contracts admit an *agreement*.

A crucial observation of [6] is that if contracts admit an agreement, then some participant is not protected, and *vice-versa*. The archetypical example is the one outlined above. Intuitively, if each participant waits until someone else has performed her action, then everyone is protected, but the contracts do not admit an agreement because of the deadlock. Otherwise, if a participant does her action without waiting, then the contracts admit an agreement, but the participant who makes the first step is not protected. This is similar to the proof of impossibility of fair exchange protocols without a trusted third party [13].

To overcome this problem, we introduce *lending Petri nets* (in short, LPN). Roughly, an LPN is a Petri net where some places may give tokens "on credit". Technically, when a place gives a token on credit its marking will become negative. This differs from standard Petri nets, where markings are always nonnegative. The intuition is that if a participant takes a token on credit, then she is obliged to honour it — otherwise she is culpable of a contract violation.

Differently from the Petri nets used in [21], LPNs allow for modeling contracts which, at the same time, admit an agreement (more formally, *weakly terminate*) and protect their participants. LPNs preserve one of the main results of [21], i.e. the possibility of proving that an application respects a choreography, by only locally verifying the services which compose it. More precisely, we project a choreography to a set of local views, independently refine each of them, and be guaranteed then the composition of all refinements respects the choreography. This is stated formally in Theorem 8.

The other main contribution is a relation between the logical contracts of [7] and LPN contracts. More precisely, we consider contracts expressed in (a fragment of) Propositional Contract Logic (PCL), and we compile them into LPNs. Theorem 23 states that a PCL contract admits an agreement if and only if its compilation weakly terminates. Summing up, Theorem 24 states that one can start from a choreography represented as a logical contract, compile it to a physical one, and then use Theorem 8 to project it to a set services which correctly implement it, and which are protected against adversaries. Finally, Theorem 25 relates logical and physical characterizations of *urgent* actions, *i.e.* those actions which must be performed in a given state of the contract.

2 Nets

We briefly review Petri nets [19] and the token game. We consider Petri nets labeled on a set $\mathcal{T}$, and (perhaps a bit unusually) the labeling is also on places.

A *labeled Petri net* is a 5-tuple $\langle S, T, F, \Gamma, \Lambda \rangle$, where S is a set of *places*, and T is a set of *transitions* (with $S \cap T = \emptyset$), $F \subseteq (S \times T) \cup (T \times S)$ is the *flow relation*, and $\Gamma : S \to \mathfrak{T}$, $\Lambda : T \to \mathfrak{T}$ are partial *labeling function* for places and transitions, respectively. Ordinary (non labeled) Petri nets are those where the two labeling functions are always undefined (*i.e.* equal to $\perp$). We require that for each $t \in T$, $F(t, s) > 0$ for some place $s \in S$, i.e. a transition cannot happen *spontaneously*. Subscripts on the net name carry over the names of the net components. As usual, we define the *pre-set* and *post-set* of a transition/place: ${}^\bullet x = \{y \in T \cup S \mid F(y, x) > 0\}$ and $x^\bullet = \{y \in T \cup S \mid F(x, y) > 0\}$, respectively. These are extended to subsets of transitions/places in the obvious way.

A *marking* is a function m from places to natural numbers (*i.e.* a multiset over places), which represents the state of the system modeled by the net. A *marked* Petri net is a pair $N = (\langle S, T, F, \Gamma, \Lambda \rangle, m_0)$, where $\langle S, T, F, \Gamma, \Lambda \rangle$ is a labelled Petri net, and $m_0 : S \to \mathbb{N}$ is the *initial marking*.

The dynamic of a net is described by the execution of transitions at markings. Let N be a marked net (hereafter we will just call net a marked net). A transition t is enabled at a marking m if the places in the pre-set of t contains enough tokens (*i.e.* if m *contains* the pre-set of t). Formally, $t \in T$ is *enabled* at m if $m(s) \geq F(s, t)$ for all $s \in {}^\bullet t$. In this case, to indicate that the execution of t in m produces the new marking $m'(s) = m(s) - F(s, t) + F(t, s)$, we write $m \,[t\rangle\, m'$, and we call it a *step*[1]. This notion is lifted, as usual, to multisets of transitions.

The notion of step leads to that of *execution* of a net. Let $N = (\langle S, T, F, \Gamma, \Lambda \rangle, m_0)$ be a net, and let m be a marking. The *firing sequences* starting at m are defined as follows: (a) m is a firing sequence, and (b) if $m \,[t_1\rangle\, m_1 \cdots m_{n-1} \,[t_n\rangle\, m_n$ is a firing sequence and $m_n \,[t\rangle\, m'$ is a step, then $m \,[t_1\rangle\, m_1 \cdots m_{n-1} \,[t_n\rangle\, m_n \,[t\rangle\, m'$ is a firing sequence. A marking m is *reachable* iff there exists a firing sequence starting at m_0 leading to it. The set of reachable markings of a net N is denoted with $\mathsf{M}(N)$. A net $N = (\langle S, T, F, \Gamma, \Lambda \rangle, m_0)$ is *safe* when each marking $m \in \mathsf{M}(N)$ is such that $m(s) \leq 1$ for all $s \in S$.

A *trace* can be associated to each firing sequence, which is the word on $\mathfrak{T}^*$ obtained by the firing sequence considering just the (labels of the) transitions and forgetting the markings: if $m_0 \,[t_1\rangle\, m_1 \cdots m_{n-1} \,[t_n\rangle\, m_n$ is a firing sequence of N, the associated trace is $\Lambda(t_1 t_2 \ldots t_n)$. The trace associated to m_0 is the empty word ε. If the label of a transition is undefined then the associated word is the empty one. The traces of a net N are denoted with $\mathit{Traces}(N)$.

A *subnet* is a net obtained by restricting places and transitions of a net, and correspondingly the flow relation and the initial marking. Let $N = (\langle S, T, F, \Gamma, \Lambda \rangle, m_0)$ be a net, and let $T' \subseteq T$. We define the subnet generated by T' as the net $N|_{T'} = (\langle S', T', F', \Gamma', \Lambda' \rangle, m_0')$, where $S' = \{s \in S \mid F(t, s) > 0 \text{ or } F(s, t) > 0 \text{ for } t \in T'\} \cup \{s \in S \mid m_0(s) > 0\}$, F' is the flow relation restricted to S' and T', Γ' is obtained by Γ restricting to places in S', Λ' is obtained by Λ restricting to transitions in T', and m_0' is obtained by m_0 restricting to places in S'

[1] The word step is usually reserved to the execution of a subset of transitions, but here we prefer to stress the computational interpretation.

A net property (intuitively, a property of the system modeled as a Petri net) can be characterized in several ways, *e.g.* as a set of markings (states of the system). The following captures the intuition that, notwithstanding the state (marking) reached by the system, it is always possible to reach a state satisfying the property. A net N *weakly terminates in* a set of markings $\mathcal{M}$ iff $\forall m \in \mathsf{M}(N)$, there is a firing sequence starting at m and leading to a marking in $\mathcal{M}$. Hereafter, we shall sometimes say that N weakly terminates (without referring to any $\mathcal{M}$) when the property is not relevant or clear from the context.

We now introduce occurrence nets. The intuition behind this notion is the following: regardless how tokens are produced or consumed, an occurrence net guarantees that each transition can occur only once (hence the reason for calling them occurrence nets). We adopt the notion proposed by van Glabbeek and Plotkin in [22], namely 1-occurrence nets. For a multiset M, we denote by $[\![M]\!]$ the multiset defined as $[\![M]\!](a) = 1$ if $M(a) > 0$ and $[\![M]\!](a) = 0$ otherwise. A *state* of a net $N = (\langle S, T, F, \Gamma, \Lambda\rangle, m_0)$ is any finite multiset X of T such that the function $m_X : S \to \mathbb{Z}$ given by $m_X(s) = m_0(s) + \sum_{t \in T} X(t) \cdot (F(t,s) - F(s,t))$, for all $s \in S$, is a reachable marking of the net. We denote by $\mathsf{St}(N)$ the states of N. A state contains (in no order) all the occurrence of the transitions that have been fired to reach a marking. Observe that a trace of a net is a suitable linearization of the elements of a state X. We use the notion of state to formalize occurrence nets. An *occurrence net* $O = (\langle S, T, F, \Gamma, \Lambda\rangle, m_0)$ is a net where each state is a set, i.e. $\forall X \in \mathsf{St}(N).\ X = [\![X]\!]$.

A net is *correctly labeled* iff $\forall s. \forall t, t' \in {}^\bullet s.\ \Gamma(s) \neq \bot \implies \Lambda(t) = \Lambda(t') = \Gamma(s)$. Intuitively, this requires that all the transitions putting a token in a labeled place represent the same action.

3 Nets with lending places

We now relax the conditions under which transitions may be executed, by allowing a transition to consume tokens from a place s even if the s does not contain enough tokens. Consequently, we allow markings with negative numbers. When the number of tokens associated to a place becomes negative, we say that they have been done *on credit*. We do not permit this to happen in all places, but only in the *lending* places (a subset $\mathcal{L}$ of S). Lending places are depicted with a double circle.

Definition 1. *A* lending *Petri net (LPN) is a triple* $(\langle S, T, F, \Gamma, \Lambda\rangle, m_0, \mathcal{L})$ *where* $(\langle S, T, F, \Gamma, \Lambda\rangle, m_0)$ *is a marked Petri net, and* $\mathcal{L} \subseteq S$ *is the set of* lending *places.*

Example 1. Consider the LPN N_1 in Fig. 1. The places p_2 and p_4 are lending places. The set of labels of the transitions is $\mathcal{T} = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$, and the set of labels of the places is $\mathcal{G} = \mathcal{T}$. The labeling is $\Gamma(p_1) = \mathsf{c}, \Gamma(p_2) = \mathsf{a}$ and $\Gamma(p_4) = \Gamma(p_3) = \mathsf{b}$ (the place p_0 is unlabeled).

The notion of step is adapted to take into account this new kind of places. Let N be an LPN, let t be a transition in T, and let m be a marking. We say that

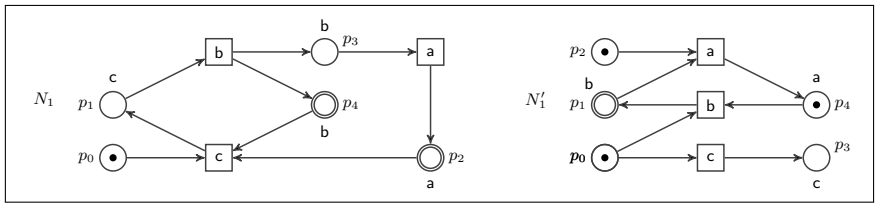

Fig. 1. Two lending Petri nets.

t is *enabled* at m iff $\forall s \in {}^\bullet t.\ m(s) \leq 0 \implies s \in \mathcal{L}$. The evolution of N is defined as before, with the difference that the obtained marking is now a function from places to $\mathbb{Z}$ (instead of $\mathbb{N}$). This notion matches the intuition behind of lending places: we allow a transition to be executed even when some of the transitions that are a pre-requisite have not been executed yet.

Definition 2. *Let m be a reachable marking of an LPN N. We say that m is* honored *iff $m(s) \geq 0$ for all places s of N.*

An honored firing sequence is a firing sequence where the final marking is honored. Note that if the net has no lending places, then all the reachable markings are honored.

Example 2. In the net of Ex. 1, the transition c is enabled even though there are no tokens in the places p_2 and p_4 in its pre-set, as they are lending places. The other transitions are not enabled, hence at the initial marking only c may be executed (on credit). After firing c, only b can be executed. This results in putting one token in p_3 and one in p_4, hence giving back the one taken on credit. After this, only a can be executed. Upon firing c, b and a, the marking is honored. The net is clearly a (correctly labeled) occurrence net.

We now introduce a notion of composition of LPNs. The idea is that the places with a label are places in an *interface* of the net (though we do not put any limitation on such places, as done instead *e.g.* in [21]) and they never are initially marked. The labelled transitions of a net are connected with the places bearing the same label of the other.

Definition 3. *Let $N = (\langle S, T, F, \Gamma, \Lambda\rangle,\ m_0, \mathcal{L})$ and $N' = (\langle S', T', F', \Gamma', \Lambda'\rangle,\ m_0', \mathcal{L}')$ be two LPNs. We say that N, N' are* compatible *whenever (a) they have the same set of labels, (b) $S \cap S' = \emptyset$, (c) $T \cap T' = \emptyset$, (d) $m_0(s) = 1$ implies $\Gamma(s) = \bot$, and (e) $m_0'(s') = 1$ implies $\Gamma'(s') = \bot$. If N and N' are compatible, their composition $N \oplus N'$ is the LPN $(\langle \hat{S}, T \cup T', \hat{F}, \hat{\Gamma}, \hat{\Lambda}\rangle, \hat{m}_0,\ \hat{\mathcal{L}})$ in Fig. 2.*

The underlying idea of LPN composition is rather simple: the sink places in a net bearing a label of a transition of the other net are removed, and places and transitions with the same label are connected accordingly (the removed sink places have places with the same label in the other net). All the other ingredients of the compound net are trivially inherited from the components. Observe that,

$$
\begin{array}{lcl}
\hat{S} & = & (S \setminus \{s \in S \mid \Gamma(s) \in \Lambda'(T') \text{ and } s^\bullet = \emptyset\}) \cup \\
 & & (S' \setminus \{s' \in S' \mid \Gamma'(s') \in \Lambda(T) \text{ and } s'^\bullet = \emptyset\}) \\
\\
\hat{F}(\hat{s},\hat{t}) & \iff & \big(\hat{s} = s_1 \in S \ \wedge\ \hat{t} = t_1 \in T \ \wedge\ F(s_1,t_1)\big) \\
 & \vee & \big(\hat{s} = s_2 \in S' \ \wedge\ \hat{t} = t_2 \in T' \ \wedge\ F'(s_2,t_2)\big) \\
\\
\hat{F}(\hat{t},\hat{s}) & \iff & \big(\hat{s} = s_1 \in S \ \wedge\ \hat{t} = t_1 \in T \ \wedge\ F(t_1,s_1)\big) \\
 & \vee & \big(\hat{s} = s_2 \in S' \ \wedge\ \hat{t} = t_2 \in T' \ \wedge\ F'(t_2,s_2)\big) \\
 & \vee & \big(\hat{s} = s \in S \ \wedge\ \hat{t} = t' \in T' \ \wedge\ \Lambda'(t') = \Gamma(s) \neq \bot\big) \\
 & \vee & \big(\hat{s} = s' \in S' \ \wedge\ \hat{t} = t \in T \ \wedge\ \Lambda(t) = \Gamma'(s') \neq \bot\big) \\
\\
\hat{\Gamma}(\hat{s}) & = & \begin{cases} \Gamma(s_1) & \textit{if } \hat{s} = s_1 \in S \\ \Gamma'(s_2) & \textit{if } \hat{s} = s_2 \in S' \end{cases} \\
\\
\hat{\Lambda}(\hat{t}) & = & \begin{cases} \Lambda(t_1) & \textit{if } \hat{t} = t_1 \in T \\ \Lambda'(t_2) & \textit{if } \hat{t} = t_2 \in T' \end{cases} \\
\\
\hat{m}_0(\hat{s}) & = & \begin{cases} 1 & \text{if } \hat{s} = s_1 \in S \text{ and } m_0(s_1) = 1, \text{ or } \hat{s} = s_2 \in S' \text{ and } m'_0(s_2) = 1 \\ 0 & \text{otherwise} \end{cases} \\
\\
\hat{\mathcal{L}} & = & (\mathcal{L} \cup \mathcal{L}') \cap \hat{S}
\end{array}
$$

Fig. 2. Composition of two LPNs.

when composing two compatible nets N and N' such that $\Gamma(S) \cap \Gamma'(S') = \emptyset$, we obtain the disjoint union of the two nets. Further, if the common label $a \in \Gamma(S) \cap \Gamma'(S')$ is associated in N to a place s with empty post-set and in N' to a place s' with empty post-set (or *vice versa*) and the labelings are injective, we obtain precisely the composition defined in [21]. If the components N and N' may satisfy some properties (sets of markings $\mathcal{M}$ and $\mathcal{M}'$), the compound net $N \oplus N'$ may satisfy the compound property (which is the set of markings $\hat{\mathcal{M}}$ obtained obviously from $\mathcal{M}$ and $\mathcal{M}'$).

Example 3. Consider the nets in Fig. 3. Net N fires a after b has been performed; dually, net N' waits for b before firing a. These nets model two participants which protect themselves by waiting the other one to make the first step (the properties being that places p_3 and p'_3, respectively, are not marked). Clearly, no agreement is possible in this scenario. This is modelled by the deadlock in the composition $N \oplus N'$, where neither transitions a nor b can be fired. Consider now the LPN N'', which differs from N only for the lending place p''_1. This models a participant which may fire a on credit, under the *guarantee* that the credit will be eventually honoured by the other participant performing b (hence, the participant modeled by N'' is still protected), and the property is then place p''_3 unmarked and p''_1 with a non negative marking. The composition $N'' \oplus N'$ weakly terminates wrt the above properties, because transition a can take a token on credit from p''_1, and then transition b can be fired, so honouring the debit in p''_1.

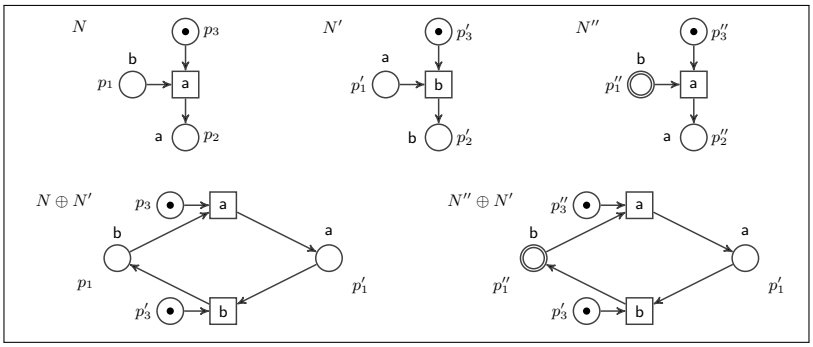

Fig. 3. Three LPNs (top) and their pairwise compositions (bottom).

The operation $\oplus$ is clearly associative and commutative.

Proposition 4 *Let N_1, N_2 and N_3 be three compatible LPNs. Then, $N_1 \oplus N_2 = N_2 \oplus N_1$ and $N_1 \oplus (N_2 \oplus N_3) = (N_1 \oplus N_2) \oplus N_3$.*

The composition $\oplus$ does not have the property that, in general, considering only the transitions of one of the components, we obtain the LPN we started with, *i.e.* $(N_1 \oplus N_2)|_{T_i} \neq N_i$. This is because the number of places with labels increases and new arcs may be added, and these places are not *forgotten* when considering the subnet generated by T_i. However these added places are not initially marked, hence it may be that the nets have the same traces.

Definition 5. *Let N and N' two LPNs on the same sets of labels. We say that N* approximates *N' ($N \lesssim N'$) iff Traces$(N) \subseteq$ Traces(N'). We write $N \sim N'$ when $N \lesssim N'$ and $N' \lesssim N$.*

Proposition 6 *For two compatible LPNs N_1, N_2, $N_i \sim (N_1 \oplus N_2)|_{T_i}$, $i = 1, 2$.*

Following [21] we introduce a notion of refinement (called *accordance* in [21]) between two LPNs. We say that M (with a property $\mathcal{M}_M$) is a *strategy* for an LPN N (with a property $\mathcal{M}$) if $N \oplus M$ is weakly terminating. With $\mathcal{S}(N)$ we denote the set of all strategies for N. In the rest of the paper we assume that properties are always specified, even when not done explicitly.

Definition 7. *An LPN N'* refines *N if $\mathcal{S}(N') \supseteq \mathcal{S}(N)$.*

Observe that if N' refines N and N weakly terminates, then N' weakly terminates as well.

If a weakly terminating LPN N is obtained by composition of several nets, *i.e.* $N = \bigoplus_i N_i$, we can ask what happens if there is an N_i' which refines N_i, for each i. The following theorem gives the desired answer.

Theorem 8 *Let $N = \bigoplus_i N_i$ be a weakly terminating LPN, and assume that N_i' refines N_i, for all i. Then, $N' = \bigoplus_i N_i'$ is a weakly terminating LPN.*

The theorem above gives a compositional criterion to check weak termination of a SOC application. One starts from an abstract specification (e.g. a choreography), projects it into a set of local views, and then refines each of them into a service implementation. These services can be verified independently (for refinement), and it is guaranteed that their composition still enjoys weak termination.

We now define, starting from a marking m, which actions may be performed immediately after, while preserving the ability to reach an honored marking. We call these actions *urgent*.

Definition 9. *For an LPN N and marking m, we say* a urgent at m *iff there exists a firing sequence $m\,[t_1\rangle \cdots [t_n\rangle\, m_n$ with $\Lambda(t_1) = \mathsf{a}$ and m_n honored.*

Example 4. Consider the nets in Ex. 3. In $N'' \oplus N'$ the only urgent action at the initial marking is a, while b is urgent at the marking where p_1' is marked. In N'' there are no urgent actions at the initial marking, since no honored marking is reachable. In the other nets (N, N', $N \oplus N'$) no actions are urgent in the initial marking, since these nets are deadlocked.

4 Physical contracts

We now present a model for physical contracts based on LPNs. Let $\mathsf{a}, \mathsf{b}, \ldots \in \mathcal{T}$ be *actions*, performed by *participants* $\mathsf{A}, \mathsf{B}, \ldots \in \mathit{Part}$. We assume that actions may only be performed once. Hence, we consider a subclass of LPNs, namely occurrence nets, where all the transitions with the same label are mutually exclusive. A physical contract is an LPN, together with a set $\mathcal{A}$ of participants bound by the contract, a mapping π from actions to participants, and a set Ω modeling the states where all the participant in $\mathcal{A}$ are satisfied.

Definition 10. *A* contract net $\mathcal{D}$ *is a tuple $(O, \mathcal{A}, \pi, \Omega)$, where O is an occurrence LPN $(\langle S, T, F, \Gamma, \Lambda\rangle, m_0, \mathcal{L})$ labeled on $\mathcal{T}$, $\mathcal{A} \subseteq \mathit{Part}$, $\pi : \mathcal{T} \to \mathit{Part}$, $\Omega \subseteq \wp(\mathcal{T})$ is the set of goals of the participants, and where:*

(a) $\forall s \in S.\ (m_0(s) = 1 \implies {}^\bullet s = \emptyset \ \wedge\ \Gamma(s) = \bot) \ \wedge\ (s \in \mathcal{L} \implies \Gamma(s) \in \mathcal{T})$,
(b) $\forall t \in T.\ (\forall s \in t^\bullet.\ \Lambda(t) = \Gamma(s)) \ \wedge\ (\exists s \in {}^\bullet t.\ s \notin \mathcal{L})$,
(c) $\forall t, t' \in T.\ \Lambda(t) = \Lambda(t') \implies \exists s \in {}^\bullet t \cap {}^\bullet t'.\ m_0(s) = 1$,
(d) $\pi(\Lambda(T)) \subseteq \mathcal{A}$.

The last constraint models the fact that only the participants in $\mathcal{A}$ may perform actions in $\mathcal{D}$.

Given a state X of the component O of $\mathcal{D}$, the reached marking m tells us which actions have been performed, and which tokens have been taken on credit. The *configuration* $\mu(m)$ associated to a marking m is the pair (C, Y) defined as:

- $C = \{\mathsf{a} \in \mathcal{T} \mid \exists s \in S.\ \{s\} = \bigcap_{t \in T} \{\, {}^\bullet t \mid \Lambda(t) = \mathsf{a}\}$ *and* $m(s) = 0\}$, and
- $Y = \{\mathsf{a} \in \mathcal{T} \mid \exists s \in S.\ \mathsf{a} = \Gamma(s)$ *and* $m(s) < 0\}$

The first component is the set of the labels of the transitions in X. The marking m is honored whenever the second component of $\mu(m)$ is empty.

We now state the conditions under which two contract nets can be composed. We require that an action can be performed only by one of the components (the other may *use* the tokens produced by the execution of such action).

Definition 11. *Two contracts nets* $\mathcal{D} = (O, \mathcal{A}, \pi, \Omega)$ *and* $\mathcal{D}' = (O', \mathcal{A}', \pi', \Omega')$ *are* compatible *whenever* $O \oplus O'$ *is defined and* $\mathcal{A} \cap \mathcal{A}' = \emptyset$.

The composition of $\mathcal{D}$ and $\mathcal{D}'$ is then the obvious extension of the one on LPNs:

Definition 12. *Let* $\mathcal{D} = (O, \mathcal{A}, \pi, \Omega)$ *and* $\mathcal{D}' = (O', \mathcal{A}', \pi', \Omega')$ *be two compatible contract nets. Then* $\mathcal{D} \oplus \mathcal{D}' = (O \oplus O', \mathcal{A} \cup \mathcal{A}', \pi \circ \pi', \Omega'')$ *where* $\Omega'' = \{X \cup X' \mid X \in \Omega, X' \in \Omega'\}$.

We lift the notion of weak termination to contract nets $\mathcal{D} = (O, \mathcal{A}, \pi, ok, \Omega)$. The set of markings obtained by Ω is $\mathcal{M}_\Omega = \{m \in \mathsf{M}(O) \mid \mu(m) = (C, \emptyset),\ C \in \Omega\}$. We say that $\mathcal{D}$ weakly terminates w.r.t. Ω when O weakly terminates w.r.t. $\mathcal{M}_\Omega$.

We also extend to contract nets the notion of urgent actions given for LPNs (Def. 9). Here, the set of urgent actions $\mathcal{U}_\mathcal{D}^C$ is parameterized by the set C of actions already performed.

Definition 13. *Let* $\mathcal{D}$ *be a contract net, and let* $C \subseteq \mathcal{T}$. *We define:*

$$\mathcal{U}_\mathcal{D}^C = \{\mathsf{a} \in \mathcal{T} \mid \exists Y \subseteq \mathcal{T}.\ \exists m.\ \mu(m) = (C, Y) \ \wedge\ \mathsf{a} \ \textit{ is urgent at } \ m\}$$

Example 5. Interpret the LPN N_1' in Fig. 1 as a contract net where the actions a, b, c are associated, respectively, to participants A, B, and C, and Ω is immaterial. Then, a and c are urgent at the initial marking, whereas b is not (the token borrowed from p_1 cannot be given back). In the state where a has been fired, only b is urgent; in the state where c has been fired, no actions are urgent.

5 Logical contracts

In this section we briefly review Propositional Contract Logic (PCL [7]), and we exploit it to model contracts. PCL extends intuitionistic propositional logic IPC with a connective $\twoheadrightarrow$, called *contractual implication*. Intuitively, a formula $\mathsf{b} \twoheadrightarrow \mathsf{a}$ implies a not only when b is true, like IPC implication, but also in the case that a "compatible" formula, e.g. $\mathsf{a} \twoheadrightarrow \mathsf{b}$, holds. PCL allows for a sort of "circular" assume-guarantee reasoning, hinted by $(\mathsf{b} \twoheadrightarrow \mathsf{a}) \wedge (\mathsf{a} \twoheadrightarrow \mathsf{b}) \to \mathsf{a} \wedge \mathsf{b}$, which is a theorem in PCL . We assume that the prime formulae of PCL coincide with the atoms in $\mathcal{T}$. PCL formulae, ranged over greek letters $\varphi, \varphi', \ldots$, are defined as:

$$\varphi \ ::= \ \bot \mid \top \mid \mathsf{a} \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \to \varphi \mid \varphi \twoheadrightarrow \varphi$$

Two proof systems have been presented for PCL: a sequent calculus [7], and an equivalent natural deduction system [4], the main rules of which are shown in Fig. 4. Provable formulae are contractually implied, according to rule ($\twoheadrightarrow$I1).

$$\frac{\Delta \vdash \psi}{\Delta \vdash \varphi \twoheadrightarrow \psi}\ (\twoheadrightarrow\text{I1}) \qquad \frac{\Delta \vdash \varphi' \twoheadrightarrow \psi' \quad \Delta, \varphi \vdash \varphi' \quad \Delta, \psi' \vdash \varphi \twoheadrightarrow \psi}{\Delta \vdash \varphi \twoheadrightarrow \psi}\ (\twoheadrightarrow\text{I2}) \qquad \frac{\Delta \vdash \varphi \twoheadrightarrow \psi \quad \Delta, \psi \vdash \varphi}{\Delta \vdash \psi}\ (\twoheadrightarrow\text{E})$$

Fig. 4. Natural deduction for PCL (rules for $\twoheadrightarrow$).

Rule ($\twoheadrightarrow$I2) provides $\twoheadrightarrow$ with the same weakening properties of $\to$. The crucial rule is ($\twoheadrightarrow$E), which allows for the elimination of $\twoheadrightarrow$. Compared to the rule for elimination of $\to$ in IPC, the only difference is that in the context used to deduce the antecedent φ, rule ($\twoheadrightarrow$E) also allows for using as hypothesis the consequence ψ. The decidability of the provability relation of PCL has been proved in [7], by exploiting the cut elimination property enjoyed by the sequent calculus.

To model contracts, we consider the Horn fragment of PCL, which comprises atoms, conjunctions, and non-nested (intuitionistic/contractual) implications.

Definition 14. *A* PCL *contract is a tuple* $\langle \Delta, \mathcal{A}, \pi, \Omega \rangle$*, where* Δ *is a Horn* PCL *theory,* $\mathcal{A} \subseteq \mathit{Part}$*,* $\pi : \mathcal{T} \to \mathit{Part}$ *associates each atom with a participant, and* $\Omega \subseteq \wp(\mathcal{T})$ *is the set of goals of the participants.*

The component $\mathcal{A}$ of $\mathcal{C}$ contains the participants which can promise to do something in $\mathcal{C}$. Consequently, we shall only consider PCL contracts such that if $\alpha \circ \mathsf{a} \in \Delta$, for $\circ \in \{\to, \twoheadrightarrow\}$, then $\pi(\mathsf{a}) \in \mathcal{A}$.

Example 6. Suppose three kids want to play together. Alice has a toy airplane, Bob has a bike, and Carl has a toy car. Each of the kids is willing to share his toy, but they have different constraints: Alice will lend her airplane only *after* Bob has allowed her ride his bike; Bob will lend his bike after he has played with Carl's car; Carl will lend his car if the other two kids promise to eventually let him play with their toys. Let $\pi = \{\mathsf{a} \mapsto \mathsf{A}, \mathsf{b} \mapsto \mathsf{B}, \mathsf{c} \mapsto \mathsf{C}\}$. The kids contracts are modeled as follows: $\langle \mathsf{b} \to \mathsf{a}, \{\mathsf{A}\}, \pi, \{\{\mathsf{b}\}\} \rangle$, $\langle \mathsf{c} \to \mathsf{b}, \{\mathsf{B}\}, \pi, \{\{\mathsf{c}\}\} \rangle$, and $\langle (\mathsf{a} \wedge \mathsf{b}) \twoheadrightarrow \mathsf{c}, \{\mathsf{C}\}, \pi, \{\{\mathsf{a}, \mathsf{b}\}\} \rangle$.

A contract admits an *agreement* when all the involved participants can reach their goals. This is formalized in Def. 15 below.

Definition 15. *A* PCL *contract admits an* agreement *iff* $\exists X \in \Omega.\ \Delta \vdash \bigwedge X$*.*

We now define composition of PCL contracts. If $\mathcal{C}'$ is the contract of an adversary of $\mathcal{C}$, then a naïve composition of the two contracts could easily lead to an attack, e.g. when Mallory's contract says that Alice is obliged to give him her airplane. To prevent from such kinds of attacks, contract composition is a partial operation. We do *not* compose contracts which bind the same participant, or which disagree on the association between atoms and participants.

Definition 16. *Two* PCL *contracts* $\mathcal{C} = \langle \Delta, \mathcal{A}, \pi, \Omega \rangle$ *and* $\mathcal{C}' = \langle \Delta', \mathcal{A}', \pi', \Omega' \rangle$ *are* compatible *whenever* $\mathcal{A} \cap \mathcal{A}' = \emptyset$*, and* $\forall \mathsf{A} \in \mathcal{A} \cup \mathcal{A}'.\ \pi^{-1}(\mathsf{A}) = \pi'^{-1}(\mathsf{A})$*. If* $\mathcal{C}$*,* $\mathcal{C}'$ *are compatible, the contract* $\mathcal{C} \mid \mathcal{C}' = \langle \Delta \cup \Delta', \mathcal{A} \cup \mathcal{A}', \pi \circ \pi', \Omega \mid \Omega' \rangle$*, where* $\Omega \mid \Omega' = \{X \cup X' \mid X \in \Omega,\ X' \in \Omega'\}$*, is their composition.*

$$\frac{}{\varepsilon \in [\![\Delta]\!]}\,(\varepsilon) \qquad \frac{\alpha \to \mathsf{a} \in \Delta \quad \sigma \in [\![\Delta]\!] \quad \overline{\alpha} \subseteq \overline{\sigma}}{\sigma\,\mathsf{a} \in [\![\Delta]\!]}\,(\to) \qquad \frac{\alpha \twoheadrightarrow \mathsf{a} \in \Delta \quad \sigma \in [\![\Delta, \mathsf{a}]\!] \quad \overline{\alpha} \subseteq \overline{\sigma}}{\sigma \mid \mathsf{a} \subseteq [\![\Delta]\!]}\,(\twoheadrightarrow)$$

Fig. 5. Proof traces of Horn PCL.

Example 7. The three contracts in Ex. 6 are compatible, and their composition is $\mathcal{C} = \langle \Delta, \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}, \{\mathsf{a} \mapsto \mathsf{A}, \mathsf{b} \mapsto \mathsf{B}, \mathsf{c} \mapsto \mathsf{C}\}, \{\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}\}\rangle$ where Δ is the theory $\{\mathsf{b} \to \mathsf{a},\ \mathsf{c} \to \mathsf{b},\ (\mathsf{a} \wedge \mathsf{b}) \twoheadrightarrow \mathsf{c}\}$. $\mathcal{C}$ has an agreement, since $\Delta \vdash \mathsf{a} \wedge \mathsf{b} \wedge \mathsf{c}$. The agreement exploits the fact that Carl's contract allows the action c to happen "on credit", before the other actions are performed.

We now recap from [4] the notion of *proof traces*, i.e. the sequences of atoms respecting the order imposed by proofs in PCL. Consider e.g. rule $(\to\text{E})$:

$$\frac{\Delta \vdash \alpha \to \mathsf{a} \qquad \Delta \vdash \alpha}{\Delta \vdash \mathsf{a}}\,(\to\text{E})$$

The rule requires a proof of all the atoms in α in order to construct a proof of a. Accordingly, if σ is a proof trace of Δ, then σa if a proof trace of Δ. Instead, in the rule $(\twoheadrightarrow\text{E})$, the antecedent α needs not necessarily be proved before a: it suffices to prove α by taking a as hypothesis.

Definition 17 (Proof traces [4]). *For a Horn* PCL *theory Δ, we define the set of* proof traces $[\![\Delta]\!]$ *by the rules in Fig. 5, where for $\sigma, \eta \in E^*$ we denote with $\overline{\sigma}$ the set of atoms in σ, with $\sigma\eta$ the concatenation of σ and η, and with $\sigma \mid \eta$ the interleavings of σ and η. We assume that both concatenation and interleaving remove duplicates from the right, e.g. $aba \mid ca = ab \mid ca = \{abc, acb, cab\}$.*

The set $\mathcal{U}^X_{\mathcal{C}}$ in Def. 18 contains, given a set X of atoms, the atoms which may be proved immediately after, following some proof trace of $\mathcal{C}$.

Definition 18 (Urgent actions [4]). *For a contract $\mathcal{C} = \langle \Delta, \ldots \rangle$ and a set of atoms X, we define $\mathcal{U}^X_{\mathcal{C}} = \{\mathsf{a} \notin X \mid \exists \sigma, \sigma'.\ \overline{\sigma} = X \wedge \sigma\,\mathsf{a}\,\sigma' \in [\![\Delta, X]\!]\}$.*

Example 8. For the contract $\mathcal{C}$ specified by the theory $\Delta = \mathsf{a} \to \mathsf{b},\ \mathsf{b} \twoheadrightarrow \mathsf{a}$, we have $[\![\Delta]\!] = \{\varepsilon, \mathsf{ab}\}$, and $\mathcal{U}^{\emptyset}_{\Delta} = \{\mathsf{a}\}$, $\mathcal{U}^{\{\mathsf{a}\}}_{\Delta} = \{\mathsf{b}\}$, $\mathcal{U}^{\{\mathsf{b}\}}_{\Delta} = \{\mathsf{a}\}$, and $\mathcal{U}^{\{\mathsf{a},\mathsf{b}\}}_{\Delta} = \emptyset$.

6 From logical to physical contracts

In this section we show, starting from a logical contract, how to construct a physical one which preserves the agreement property. Technically, we shall relate provability in PCL to reachability of suitable configurations in the associated LPN. The idea of our construction is to translate each Horn clause of a PCL formula into a transition of an LPN, labelled with the action in the conclusion of the clause.

$$
\begin{aligned}
S &= (\mathcal{T} \times T) \cup \big((\{\mathsf{a} \mid \bigwedge X \to \mathsf{a} \in \Delta\} \cup \{\mathsf{a} \mid \bigwedge X \twoheadrightarrow \mathsf{a} \in \Delta\} \cup \{\mathsf{a} \mid \mathsf{a} \in \Delta\}) \times \{*\}\big) \\
T &= \{(X, \mathsf{a}, \bigcirc) \mid \bigwedge X \to \mathsf{a} \in \Delta\} \cup \{(X, \mathsf{a}, \circledcirc) \mid \bigwedge X \twoheadrightarrow \mathsf{a} \in \Delta\} \\
F &= \{(s,t) \mid s = (\mathsf{a}, *),\ t = (X, \mathsf{a}, z)\} \cup \{(s,t) \mid s = (\mathsf{a}, t),\ t = (X, \mathsf{c}, z),\ \mathsf{a} \in X\} \cup \\
&\quad \{(t,s) \mid s = (\mathsf{a}, x),\ t = (X, \mathsf{a}, z),\ x \neq *\} \\
\Gamma(s) &= \text{if } s = (\mathsf{a}, x) \text{ with } x \in T \text{ then } \mathsf{a} \text{ else } \bot \\
\Lambda(t) &= \text{if } t = (X, \mathsf{a}, z) \text{ then } \mathsf{a} \text{ else } \bot \\
m_0(s) &= \text{if } s = (\mathsf{a}, *) \text{ then 1 else 0} \\
\mathcal{L} &= \{s \in S \mid s = (\mathsf{a}, t) \textit{ and } t = (X, \mathsf{c}, \circledcirc) \textit{ with } X \neq \emptyset\}
\end{aligned}
$$

Fig. 6. Translation from logical to physical contracts.

Definition 19. *Let $\mathcal{C} = \langle \Delta, \mathcal{A}, \pi, \Omega \rangle$ be a* PCL *contract. We define the contract net $\mathcal{P}(\mathcal{C})$ as $((\langle S, T, F, \Gamma, \Lambda \rangle, m_0, \mathcal{L}), \mathcal{A}, \pi, \Omega)$ in Fig. 6.*

The transitions associated to $\mathcal{C}$ are a subset T of $\wp(\mathcal{T}) \times \mathcal{T} \times \{\circledcirc, \bigcirc\}$. For each intuitionistic/contractual implication, we introduce a transition as follows. A clause $\bigwedge X \twoheadrightarrow \mathsf{a}$ maps to $(X, \mathsf{a}, \circledcirc) \in T$, while $\bigwedge X \to \mathsf{a}$ maps to $(X, \mathsf{a}, \bigcirc) \in T$. A formula a is dealt with as the clause $\bigwedge \emptyset \to \mathsf{a}$. Places in S carry the information on which transition may actually put/consume a token from them (even on credit). The lending places are those places (a, t) where $t = (X, \mathsf{c}, \circledcirc)$. Observe that a transition $t = (X, \mathsf{a}, z)$ puts a token in each place (a, x) with $x \neq *$, and all the transitions bearing the same labels, say a, are mutually excluding each other, as they share the unique input place $(\mathsf{a}, *)$. The initial marking will contains all the places in $\mathcal{T} \times \{*\}$, and if a token is consumed from one of these places then the place will be never marked again. Furthermore the lending places are never initially marked.

Example 9. Consider the PCL contract with formula $\mathsf{a} \twoheadrightarrow \mathsf{a}$ (the other components are immaterial for the sake of the example). The associated LPN is in Fig. 7, left. The transition $(\{\mathsf{a}\}, \mathsf{a}, \circledcirc))$, labeled a, can be executed at the initial marking, as the unmarked place in the preset is a lending place. The reached marking contains no tokens, hence it is honored. This is coherent with the fact that $\mathsf{a} \twoheadrightarrow \mathsf{a} \vdash \mathsf{a}$ holds in PCL.

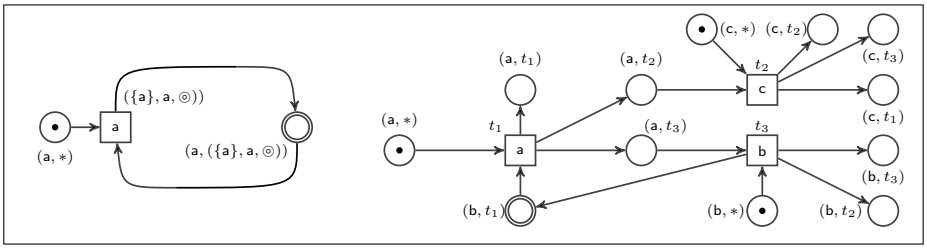

Fig. 7. Two contract nets constructed from PCL contracts.

Example 10. Consider the PCL contract specified by the theory

$$\Delta \;=\; \{\mathsf{b} \twoheadrightarrow \mathsf{a},\ \mathsf{a} \to \mathsf{c},\ \mathsf{a} \to \mathsf{b}\}$$

The associated LPN is the one on the right depicted in Fig. 7. The transitions are $t_1 = (\{\mathsf{b}\}, \mathsf{a}, \circledcirc)$, $t_2 = (\{\mathsf{a}\}, \mathsf{c}, \bigcirc)$ and $t_3 = (\{\mathsf{a}\}, \mathsf{b}, \bigcirc)$. Initially only t_1 is enabled, lending a token from place (b, t_1). This leads to a marking where both t_2 and t_3 are enabled, but only the execution of t_3 ends up with an honored marking. The marking reached after executing all the actions is honored. This is coherent with the fact that $\Delta \vdash \mathsf{a} \wedge \mathsf{b} \wedge \mathsf{c}$ holds in PCL.

Since all the transitions consume the token from the places $(\mathsf{a}, *)$ (where a is the label of the transition), and these places cannot be marked again, it is easy to see that each transition may occur only once. Hence, the net associated to a contract is an occurrence net. If two transitions t, t' have the same label (say a), then they cannot belong to the same state of the net. In fact, transitions with the same label share the same input place $(\mathsf{a}, *)$. This place is not a lending one, and has no ingoing arcs, hence only one of the transitions with the same label may happen. The notion of correctly labeled net lifts obviously to contract nets.

Proposition 20 *For all* PCL *contracts* $\mathcal{C}$*, the net* $\mathcal{P}(\mathcal{C})$ *is correctly labeled.*

A relevant property of $\mathcal{P}$ is that it is an homomorphism with respect to contracts composition. Thus, since both $\mid$ and $\oplus$ are associative and commutative, we can construct a physical contract from a set of logical contracts $\mathcal{C}_1 \cdots \mathcal{C}_n$ componentwise, i.e. by composing the contract nets $\mathcal{P}(\mathcal{C}_1) \cdots \mathcal{P}(\mathcal{C}_n)$.

Proposition 21 *For all* $\mathcal{C}_1, \mathcal{C}_2$*, we have that* $\mathcal{P}(\mathcal{C}_1 \mid \mathcal{C}_2) \sim \mathcal{P}(\mathcal{C}_1) \oplus \mathcal{P}(\mathcal{C}_2)$.

In Theorem 23 below we state the main result of this section, namely that our construction maps the agreement property of PCL contracts into weak termination of the associated contract nets. To prove Theorem 23, we exploit the fact that C is a set of provable atoms in the logic iff $(C, \emptyset)$ is a configuration of the associated contract net.

Lemma 22 *Let* $\mathcal{C} = \langle \Delta, \mathcal{A}, \pi, \Omega \rangle$ *be a* PCL *contract, and let* $\mathcal{P}(\mathcal{C}) = (O, \mathcal{A}, \pi, \Omega)$*. For all* $C \subseteq \mathcal{T}$*,* $\Delta \vdash \bigwedge C$ *iff there exists* $m \in \mathsf{M}(O)$ *such that* $\mu(m) = (C, \emptyset)$*.*

Theorem 23 $\mathcal{C}$ *admits an agreement iff* $\mathcal{P}(\mathcal{C})$ *weakly terminates in* Ω*.*

We now specialize Theorem 8, which allows for compositional verification of choreographies. Assuming a choreography specified as a PCL contract $\mathcal{C}$, we can (i) project it into the contracts $\mathcal{C}_1 \cdots \mathcal{C}_n$ of its participants, (ii) construct the corresponding LPN contracts $\mathcal{P}(\mathcal{C}_1) \cdots \mathcal{P}(\mathcal{C}_n)$, and (iii) individually refine each of them into a service implementation. If the original choreography admits an agreement, then the composition of the services weakly terminates, i.e. it is correct w.r.t. the choreography.

Theorem 24 *Let $\mathcal{C} = \mathcal{C}_1 \mid \cdots \mid \mathcal{C}_n$ admit an agreement, with Ω_i goals of $\mathcal{C}_i$. If $\mathcal{D}_i$ refines $\mathcal{P}(\mathcal{C}_i)$ for $i \in 1..n$, then $\mathcal{D}_1 \oplus \cdots \oplus \mathcal{D}_n$ weakly terminates in $\Omega_1 \cup \cdots \cup \Omega_n$.*

The notion of urgency in contract nets correspond to that in the associated PCL contracts (Theorem 25).

Theorem 25 *For all* PCL *contracts $\mathcal{C}$, and for all $X \subseteq \mathcal{T}$, $\mathcal{U}^X_{\mathcal{C}} = \mathcal{U}^X_{\mathcal{P}(\mathcal{C})}$.*

Example 11. Recall from Ex. 8 that, for $\mathcal{C} = \langle \{\mathsf{a} \to \mathsf{b}, \mathsf{b} \twoheadrightarrow \mathsf{a}\}, \ldots \rangle$, we have:

$$\mathcal{U}^{\emptyset}_{\mathcal{C}} = \{\mathsf{a}\} \qquad \mathcal{U}^{\{\mathsf{a}\}}_{\mathcal{C}} = \{\mathsf{b}\} \qquad \mathcal{U}^{\{\mathsf{b}\}}_{\mathcal{C}} = \{\mathsf{a}\} \qquad \mathcal{U}^{\{\mathsf{a},\mathsf{b}\}}_{\mathcal{C}} = \emptyset$$

This is coherent with the fact that, in the corresponding contract net $N'' \oplus N'$ in Fig. 3, only a is urgent at the initial marking, while b becomes urgent after a has been fired.

7 Related work and conclusions

We have investigated how to compile logical into physical contracts. The source of the compilation is the Horn fragment of Propositional Contract Logic [7], while the target is a contract model based on lending Petri nets (LPNs). Our compilation preserves agreements (Theorem 23), as well as the possibility of protecting services against misbehavior of malevolent services. LPN contracts can be used to reason compositionally about the realization of a choreography (Theorem 24), so extending a result of [21]. Furthermore, we have given a logical characterization of those *urgent* actions which have to be performed in a given state. This notion, which was only intuitively outlined in [7], is now made formal through our compilation into LPNs (Theorem 25).

Contract nets seem a promising model for reasoning on contracts: while having a clear relation with PCL contracts, they may inherit as well the whole realm of tools that are already available for Petri nets.

The notion of places with a negative marking is not a new one in the Petri nets community, though very few papers tackle this notion, as the interpretation of *negative* tokens does not match the intuition of Petri nets, where tokens are generally intended as resources. In this paper we have used negative tokens to model situations where actions are in a *circular* dependency, like the ones arising in PCL contracts. Lending places model the intuition that an action can be performed on a *promise*, and a negative token in a place can be interpreted as the promise made, which must be, sooner or later, *honored*. Indeed, the net obtained from a PCL contract is an occurrence net which may contain cycles, *e.g.* in the net of Ex. 10 the transition t_1 depends on t_3, which in turn depends on t_1 (and to execute t_1 we required to *lend* a token which is after supplied by t_3). In [20] the idea of places with negative marking is realized using a new kind of arc, called *debit* arcs. Under suitable conditions, these nets are Turing powerful, whereas our contract nets do not add expressiveness (while for LPNs the issue has to be investigated). In [17] negative tokens arise as the result of certain *linear* assumptions. The relations with LPNs have to be investigated.

Acknowledgments. We thank Philippe Darondeau, Eric Fabre and Roberto Zunino for useful discussions and suggestions. This work has been partially supported by Aut. Reg. of Sardinia grants L.R.7/2007 CRP2-120 (TESLA) CRP-17285 (TRICS) and P.I.A. 2010 ("Social Glue"), and by MIUR PRIN 2010-11 project "Security Horizons".

References

1. M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for access control in distributed systems. *ACM TOPLAS*, 4(15), 1993.
2. M. Abadi and G. D. Plotkin. A logical view of composition. *TCS*, 114(1), 1993.
3. M. Armbrust et al. A view of cloud computing. *Comm. ACM*, 53(4):50–58, 2010.
4. M. Bartoletti, T. Cimoli, P. D. Giamberardino, and R. Zunino. Contract agreements via logic. In *Proc. ICE*, 2013.
5. M. Bartoletti, T. Cimoli, G. M. Pinna, and R. Zunino. An event-based model for contracts. In *Proc. PLACES*, 2012.
6. M. Bartoletti, T. Cimoli, and R. Zunino. A theory of agreements and protection. In *Proc. POST*, 2013.
7. M. Bartoletti and R. Zunino. A calculus of contracting processes. In *LICS*, 2010.
8. L. Bocchi, K. Honda, E. Tuosto, and N. Yoshida. A theory of design-by-contract for distributed multiparty interactions. In *CONCUR*, 2010.
9. M. Bravetti, I. Lanese, and G. Zavattaro. Contract-driven implementation of choreographies. In *Proc. TGC*, pages 1–18, 2008.
10. M. Bravetti and G. Zavattaro. Contract based multi-party service composition. In *Proc. FSEN*, pages 207–222, 2007.
11. M. Bravetti and G. Zavattaro. Towards a unifying theory for choreography conformance and contract compliance. In *Software Composition*, 2007.
12. G. Castagna, N. Gesbert, and L. Padovani. A theory of contracts for web services. *ACM Transactions on Programming Languages and Systems*, 31(5), 2009.
13. S. Even and Y. Yacobi. Relations among public key signature systems. Technical Report 175, Computer Science Department, Technion, Haifa, 1980.
14. D. Garg and M. Abadi. A modal deconstruction of access control logics. In *FoSSaCS*, 2008.
15. T. T. Hildebrandt and R. R. Mukkamala. Declarative event-based workflow as distributed dynamic condition response graphs. In *Proc. PLACES*, 2010.
16. K. Honda, N. Yoshida, and M. Carbone. Multiparty asynchronous session types. In *POPL*, 2008.
17. N. Martí-Oliet and J. Meseguer. An algebraic axiomatization of linear logic models. In *Topology and category theory in computer science*, 1991.
18. C. Prisacariu and G. Schneider. A dynamic deontic logic for complex contracts. *The Journal of Logic and Algebraic Programming (JLAP)*, 81(4), 2012.
19. W. Reisig. *Petri Nets: An Introduction*, volume 4 of *Monographs in Theoretical Computer Science. An EATCS Series*. Springer, 1985.
20. P. D. Stotts and P. Godfrey. Place/transition nets with debit arcs. *Inf. Proc. Lett.*, 41(1), 1992.
21. W. M. P. van der Aalst, N. Lohmann, P. Massuthe, C. Stahl, and K. Wolf. Multiparty contracts: Agreeing and implementing interorganizational processes. *Comput. J.*, 53(1), 2010.
22. R. J. van Glabbeek and G. D. Plotkin. Configuration structures. In *LICS*, 1995.