Science of Computer Programming 112 (2015) 75–101

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

# Lending Petri nets ☆

Massimo Bartoletti, Tiziana Cimoli, G. Michele Pinna *

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

ARTICLE INFO

*Article history:*
Received 6 June 2014
Received in revised form 14 May 2015
Accepted 18 May 2015
Available online 27 May 2015

*Keywords:*
Petri nets
Contracts
Intuitionistic Logic

ABSTRACT

We study Lending Petri nets, an extension of Petri nets where places may carry a negative number of tokens. This allows for modeling contracts where a participant may promise to give some of her resources under the guarantee that some other resources will eventually be obtained in exchange. We then propose an interpretation of the Horn fragment of Propositional Contract Logic in Lending Petri nets. In particular, we show that provability in the logic corresponds to reachability of certain markings in nets, and that proof traces correspond to "honored" firing sequences in nets.



## 1. Introduction

Service-oriented computing (SOC) and cloud computing 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, e.g., in case of mutually distrusted organizations. Thus, services must play a double role: while cooperating together, they have to protect themselves against other services 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 [1]. Quoting from [1], "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. For instance, in the approaches based on multi-party session types [2,3], *global types* are used to specify the overall behavior (i.e. the choreography) of a distributed application; a global type is then projected into *local types*, which specify the behavior expected from each service involved in the whole application. The local types can be interpreted as the service contracts: if the actual implementation of each service respects its contract, then the overall application is guaranteed to enjoy some correctness properties, e.g., deadlock freedom and session fidelity. Another approach is the bottom–up one: each service advertises its contract (a local view), and a contract broker combines those services whose contracts admit an *agreement*. This can be done, for instance, by using

☆ Work partially supported by Aut. Region of Sardinia under grant L.R.7/2007 CRP-17285 (TRICS), P.I.A. 2010 Project "Social Glue", by MIUR PRIN 2010-11 project "Security Horizons", and by EU COST Action IC1201 (BETTY).

* Corresponding author at: Dipartimento di Matematica e Informatica, Università degli Studi di Cagliari, via Ospedale 72, 09124 Cagliari (Italy).
*E-mail address:* gmpinna@unica.it (G.M. Pinna).

http://dx.doi.org/10.1016/j.scico.2015.05.006
0167-6423/

session types as contracts, and by taking as agreement the possibility to synthesize from them a choreography — i.e. a global view — whose projections are the contracts themselves [4].

Contracts may be seen as a way to formally specify and regulate the exchange of resources among the participants involved in an interaction. Typically, these resources are exchanged in a circular way: a participant provides the others with some resource, in order to obtain some others resources in return. For instance, assume that a participant A wants to obtain 1 TB disk space from a cloud storage provider B, which in turn asks a payment of \$100. We could model the contract of A as "pay \$100", and that of B as "receive \$100, and then provide A with 1 TB disk space". Intuitively, these two contracts admit an agreement: indeed, if both A and B are honest, then each one will perform its due action, so leading to a correct execution of the contracts. However, it would be unsafe for A to advertise a contract which just states "pay \$100", because this would admit an agreement also with the contract of a malicious provider which accepts the payment from A, and then gives nothing in return. To cope with this issue, A would like to advertise a stronger contract, e.g., "receive 1 TB disk space from B, and then pay \$100". However, this contract would not admit an agreement with the one of the provider: since no one is willing to do the first move, we reach a deadlock situation, where the two resources are not exchanged.

Scenarios like the one outlined above are typical in interorganizational processes, where services are mutually distrusting, and may pursue their providers goals to the detriment of the other ones. The role of contracts in these competitive scenarios is twofold: on the one hand, they must allow participants to find agreements when there is a matching between the requested and offered resources; on the other hand, they must somehow protect their participants from interactions with malicious counterparts.

Petri nets [5] offer a natural way to formalize contracts: a resource can be seen as the presence of a token in a given place, and transferring a resource can be seen as firing a transition which moves the token to another place. For instance, the contract "receive \$100, and then provide A with 1 TB disk space" can be formalized as a Petri net with two places (called, say, \$100 and 1 TB), and a transition taking one token from place \$100 and putting one token in place 1 TB. However, when composing this net with the one modeling the dual contract "receive 1 TB disk space, and then pay \$100 to B", we obtain a net which can fire no transitions, as intuitively predicted above. To overcome this deadlock situation would need to weaken the conditions under which a resource is transferred, e.g., the transition of A's contract could be fired in the absence of a token in place 1 TB, *under the guarantee* that B's transition will be eventually fired. A possible way to state this requirement is to record, after firing A's transition, that the resource \$100 has been given "on credit"; the contracts will admit an agreement only if credits will be honored, whatever the future choices of the participants.

*Contribution* In this paper we propose a model of contracts based on Petri nets, along the lines of [6]. Differently from standard Petri nets, our *Lending Petri nets* (in short, LPNs) allow places to give tokens "on credit": technically, when a place gives a token on credit, its marking may become negative (whereas markings are always non-negative in standard Petri nets). To represent contracts, we enrich LPNs with some additional information: the participants associated to each transition, and their objectives. Taking inspiration from [7], we then interpret contracts as games, where each participant has a strategy to choose which transitions to fire in order to reach her objectives.

A set of contracts admits an agreement whenever, in their composition, each participant has a winning strategy, which allows her to reach the objectives, or make some other participant *liable* of a contract violation. LPNs can model contracts which, at the same time, admit an agreement and protect their participants. In the above scenario, participant A could formalize her contract as an LPN with a transition which takes one token on credit from place 1 TB, and produces one token in place \$100. When this LPN is composed with the one of B which moves a token from \$100 to 1 TB, there is a correct exchange of resources, and so the contracts admit an agreement. Instead, when the LPN of A is composed with the one of B which just takes the token from \$100 (and gives nothing in return), there is no agreement, because the credit \$100 is not honored.

Lending Petri nets preserve one of the main results of [6], i.e., compositional verification (Theorem 3.15). More precisely, if we have an agreement among a set of contracts, then we can indepently refine each of them (e.g., into a more detailed implementation), and be guaranteed that the composition of the refined contracts still enjoys agreement.

The other main contribution of this paper is a correspondence between Lending Petri nets and a logical model of contracts, namely Propositional Contract Logic (PCL [8]). This is an extension of intuitionistic propositional logic (IPC), featuring a new binary connective $\twoheadrightarrow$, called *contractual implication*. The intuition is that, while the formula $(a \to b) \wedge (b \to a)$ in IPC is a "vicious circle", from which one can deduce neither $a$ nor $b$, the formula $(a \twoheadrightarrow b) \wedge (b \twoheadrightarrow a)$ is a "virtuous circle", which entails both $a$ and $b$. The relation with LPNs is clear: $a \to b$ is like a transition in a standard Petri net, which consumes one token from $a$ and produces a token in $b$; instead, $a \twoheadrightarrow b$ is like a transition in an LPN, which puts a token in $b$ also when the one in $a$ is missing: in such case, a later transition is required to eventually honor the credit. We exploit this insight to provide a sound and complete model of the Horn fragment of PCL. More precisely, in Definition 4.3 we associate each Horn PCL theory $\Delta$ with an LPN $\mathcal{P}(\Delta)$, and in Theorem 4.10 we show that an atom is provable in $\Delta$ if and only if a certain marking is reachable in $\mathcal{P}(\Delta)$. In Theorem 4.28 we push further the correspondence between PCL and LPNs, by showing that proof traces [9] of a Horn PCL theory $\Delta$ are exactly the honored firing sequences in $\mathcal{P}(\Delta)$.

*Structure of the paper* The rest of this paper is organized as follows. We introduce Lending Petri nets in Section 2. In Section 3 we use them as a model for contracts, and we set up a game-theoretic framework for contract agreement. In Section 4 we show that Lending Petri nets are a model of Horn PCL theories, and that honored firing sequences in LPNs correspond to