Dynamic Soundness in Resource-Constrained Workflow Nets

María Martos-Salgado, Fernando Rosa-Velardo

Dynamic Soundness in Resource-Constrained Workflow Nets*

María Martos-Salgado, and Fernando Rosa-Velardo

Sistemas Informáticos y Computación,
Universidad Complutense de Madrid
E-mail: mrmartos@estumail.ucm.es,fernandorosa@sip.ucm.es

Abstract. Workflow Petri nets (wf-nets) are an important formalism for the modeling of business processes. For them we are typically interested in the soundness problem, that intuitively consists in deciding whether several concurrent executions can always terminate properly. Resource-Constrained Workflow Nets (rcfw-nets) are wf-nets enriched with static places, that model global resources. In this paper we prove the undecidability of soundness for rcwf-nets when there may be several static places and in which instances are allowed to terminate having created or consumed resources. In order to have a clearer presentation of the proof, we define an asynchronous version of a class of Petri nets with dynamic name creation. Then, we prove that reachability is undecidable for them, and reduce it to dynamic soundness in rcwf-nets. Finally, we prove that if we restrict our class of rcwf-nets, assuming in particular that a single instance is sound when it is given infinitely many global resources, then dynamic soundness is decidable by reducing it to the home space problem in P/T nets for a linear set of markings.

1 Introduction

Workflow Nets have been identified and widely used as a solid model of business processes [1], with a rich theory for their analysis and verification, sustained in more than 40 years of development of the theory of Petri Nets. Workflow nets (wf-nets) model business processes, that start in a given initial state and must eventually finish (under a strong fairness assumption) in a final state in which its task has been completed. One of the central problems in this area is that of soundness, that of checking whether a wf-net can always reach its final state properly [2].

Here we follow the works in [3, 4]. In them, the authors study extensions of wf-nets in which processes must share some global resources. Resource-constrained workflow nets (rcwf-nets) are wf-nets in which some places are dynamic and some are static. Following a terminology from OOP, a rcwf-net can be seen as the definition of a class, with its local and static attributes, represented by

* Work supported by the MEC Spanish project DESAFIOS10 TIN2009-14599-C03-01, and Comunidad de Madrid program PROMETIDOS S2009/TIC-1465.

dynamic and static places, respectively. Then, the rcwf-net can be instantiated several times, but every instance must share the tokens in static places.

Even if a singe instance of a rcwf-net is sound, several instances could deadlock because of static places. In [3] the authors define dynamic soundness, which essentially amounts to the condition stating that any number of instances running simultaneously can always reach the final state, that in which all the tasks have been completed.

In both works, the authors consider rcwf-nets that do not create or consume static resources, that is, rcwf-nets that always return a global resource after using it. In particular, the behavior of a single instance of a rcwf-net is such that the number of tokens in the static places in the initial and final markings coincide. Under this assumption, the number of tokens in the static places is bounded by the number of tokens in the initial marking. The authors prove in [3] that dynamic soundness is decidable whenever there is only a single static place, that is, whenever there is a single type of global resources. Recently, [4] further studies the problem of dynamic soundness, extending the previous result to rcwf-nets with any number of static places, but considering a fixed number of initial resources (unlike in [3], in which the existence of a minimal number of resources for which the rcwf-net is sound is part of the problem). Under these assumptions, it is enough for the authors to study the absence of deadlocks.

In this paper we continue the works in [3, 4] by studying the problem of dynamic soundness for rcwf-nets with any number of static places, and without restricting their behavior so that instances can terminate their task having created new global resources or having consumed some.

We prove that dynamic soundness under these hypotheses is undecidable. It is to the best of our knowledge the first undecidability result regarding soundness in wf-nets (without special arcs like inhibitor or reset arcs [5]). The proof considers a class of colored Petri nets with dynamic fresh name creation that we have defined in previous works [6], called ν-PN. It is based on a non-trivial reduction of reachability in ν-PN, which is undecidable [7, 8], to dynamic soundness in rcwf-nets. Moreover, we believe that the simulation of ν-PN by means of rcwf-nets, and the reduction of the reachability problem, are interesting by themselves, and could be used to obtain other decidability/undecidability results.

Finally, we consider the same problem for a subclass of rcwf-nets, arguably a sensible subclass of rcwf-nets. We will consider rcwf-nets which are sound for a single instance (that is, such that a single instance can always finish properly) whenever it is provided with infinitely many resources, and such that every transition may contribute to the completion of the task. We will prove that dynamic soundness in this case can be reduced to a home space problem in ordinary P/T nets, which is decidable [9, 10].

The rest of the paper is organized as follows. Section 2 presents the basic concepts we will need, like P/T nets and ν-PN. Section 3 defines asynchronous ν-PN and proves undecidability of reachability for them. In Sect. 4 we present our rcwf-nets, in terms of asynchronous ν-PN. Section 5 proves that dynamic soundness is undecidable for rcwf-nets. In Sect. 6 we consider a restricted version

of the problem, and prove decidability in this case. Finally, Sect. 7 presents our conclusions and directions for further study.

2 Preliminaries

A (finite) multiset m over a set A is a mapping $m : A \to \mathbb{N}$. We denote by $A^{\oplus}$ the set of finite multisets over A. For two multisets m_1 and m_2 over A we define $m_1+m_2 \in A^{\oplus}$ by $(m_1+m_2)(a) = m_1(a)+m_2(a)$ and $m_1 \subseteq m_2$ if $m_1(a) \leq m_2(a)$ for every $a \in A$. When $m_1 \subseteq m_2$ we can define $m_2-m_1 \in A^{\oplus}$ by $(m_2-m_1)(a) = m_2(a) - m_1(a)$. We denote by $\emptyset$ the empty multiset, that is, $\emptyset(a) = 0$ for every $a \in A$. Finally, for $\lambda \in \mathbb{N}$ and $m \in A^{\oplus}$ we define $\lambda * m = m + \overset{\lambda}{\ldots} + m \in A^{\oplus}$.

Petri Nets. A Place/Transition Net [11] (P/T net for short) is a tuple $N = (P, T, F)$, where P is a finite set of places, T is a finite set of transitions (disjoint with P) and $F : (P \times T) \cup (T \times P) \to \mathbb{N}$ is the flow function. P/T nets are depicted as usual: places are drawn by circles, transitions are drawn by boxes and F is represented by arrows labeled by a natural, that is, we draw an arrow from x to y labeled by $F(x, y)$ (or without any label if $F(x, y) = 1$). We do not show the arrow whenever $F(x, y) = 0$.

A marking of N is an element of $P^{\oplus}$. For a transition t we define ${}^\bullet t \in P^{\oplus}$ as ${}^\bullet t(p) = F(p, t)$. Analogously, we take $t^\bullet(p) = F(t, p)$. A marking m enables a transition $t \in T$ if ${}^\bullet t \subseteq m$. In that case t can be fired, reaching the marking $m' = (m - {}^\bullet t) + t^\bullet$, in which case we write $m \xrightarrow{t} m'$.

Given a P/T net $N = (P, T, F)$ with initial marking m_0, we say that a place $p \in P$ is *bounded* if there exists $b \in \mathbb{N}$ such that for each reachable marking m, $m(p) \leq b$. It is bounded if all its places are bounded. Boundedness is decidable for P/T nets. Moreover, the problem of deciding whether a place is bounded, is also decidable [10]. Given a P/T net N, a marking m_0 and a set $\mathcal{H}$ of markings of N, we say that $\mathcal{H}$ is a home space if for every reachable marking m, there is a marking $m' \in \mathcal{H}$ reachable from m.

The problem of deciding whether a linear set of markings is a home space is decidable too [10, 9]. A linear set of markings of a P/T net N is a set of markings that can be obtained as linear combinations of markings of N. More precisely, a marking m_0 and a finite set of markings $\{m_1, ..., m_n\}$ define the linear set of markings $\mathcal{L} = \{m_0 + \sum_{i=1}^{n} \lambda_i * m_i \mid \lambda_i \in \mathbb{N}\}$.

Workflow Petri Nets. We will use the definition in [4]. A workflow Petri net (shortly a wf-net) is a P/T net $N = (P, T, F)$ such that:

- there are $in, out \in P$ with ${}^\bullet in = \emptyset$ and $out^\bullet = \emptyset$,
- for each $p \in P \setminus \{in, out\}$, ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$.

The second condition intuitively states that all the places contribute to the completion of the task. In this paper, we can always force that condition to be satisfied, so that we will from now on ignore it.

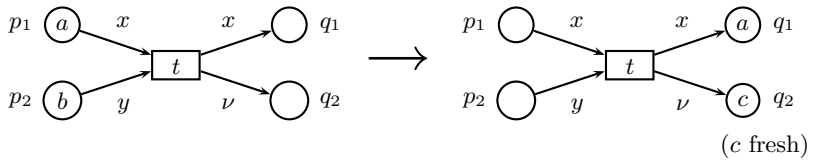

Fig. 1. Two simple ν-PN

Petri Nets with dynamic name creation. Now we briefly define ν-PN [6]. We consider an infinite set Id of names, a set Var of variables and a subset of special variables $\Upsilon \subset Var$ for name creation. A ν-PN is a tuple $N = (P, T, F)$, where P and T are finite disjoint sets, and $F : (P \times T) \cup (T \times P) \to Var^{\oplus}$.

A *marking* is a mapping $m : P \to Id^{\oplus}$. We denote by $\emptyset$ the empty marking, that which satisfies $\emptyset(p) = \emptyset$ for all $p \in P$. We write $Id(m)$ to denote the set of names that appear in m. We denote by $Var(t)$ the set of variables in arcs adjacent to t. Analogously, we will write $Var(p)$ for the set of variables adjacent to a place p. A *mode* is a mapping $\sigma : Var(t) \to Id$. A transition t can be fired with mode σ for a marking m if for all $p \in P$, $\sigma(F(p,t)) \subseteq m(p)$ and for every $\nu \in \Upsilon$, $\sigma(\nu) \notin m(p)$ for all p. In that case we have $m \xrightarrow{t} m'$, where $m'(p) = (m(p) - \sigma(F(p,t))) + \sigma(F(t,p))$ for all $p \in P$.

In order to keep usual notations in P/T nets, we will assume that there is a "distinguished" color $\bullet \in Id$. By "name" we mean any color different from $\bullet$. Moreover, we will use a distinguished variable ϵ that can only be instantiated to $\bullet$, which will be omitted in our figures. Thus, we can manage ordinary black tokens with the same notations as in P/T nets.

The reachability problem is undecidable for ν-PN [7, 8], that is, given m_0 and m_f, markings of a ν-PN N, the problem of deciding whether $m_0 \to^* m_f$ is undecidable. By following a standard reduction that removes m_f, we can prove that the problem of deciding whether the empty marking is reachable is also undecidable. Moreover, without loss of generality we can assume that m_0 contains a single token.

3 Asynchronous ν-PN

Intuitively, one can see each name in a ν-PN as a process. Then, we can see a firing of a transition in which different names are involved as a synchronization between the corresponding processes.

Next, we prove that we can assume that actually each process can only synchronize with a global shared memory, so that a synchronization between two processes must be achieved via this shared memory. Technically, we will use ordinary black tokens to represent this global memory, and names to represent processes.

Definition 1. *An asynchronous ν-PN is a ν-PN (P, T, F) such that:*

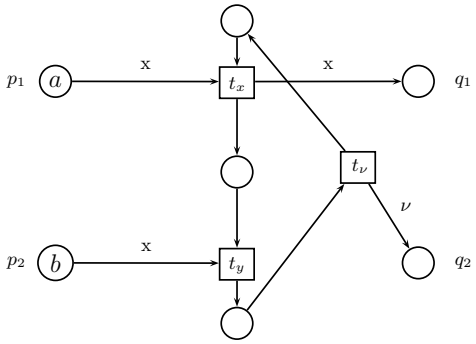

Fig. 2. Simulation of the ν-PN in the left of Fig. 1 by an asynchronous ν-PN

- *for each* $t \in T$, *either* $Var(t) \subseteq \{\nu, \epsilon\}$ *or* $Var(t) \subseteq \{x, \epsilon\}$,
- *for each* $p \in P$, *either* $Var(p) = \{x\}$ *or* $Var(p) = \{\epsilon\}$.

We call static places those $p \in P$ with $Var(p) = \{\epsilon\}$, and dynamic places those $p \in P$ with $Var(p) = \{x\}$. We will write $P = P_S \cup P_D$, with P_S the set of static places and P_D the set of dynamic places. Thus, we will disallow a situation in which $x, y \in Var(t)$. Let us now see that asynchronous ν-PN can simulate ν-PN so that reachability is preserved.

Proposition 1. *Let* N *be a* ν*-PN, and* m_0 *a marking of* N*. There is an asynchronous* ν*-PN* N' *and a marking* m'_0 *of* N' *such that* $m_0 \rightarrow^* \emptyset$ *iff* $m'_0 \rightarrow^* \emptyset$*.*

Proof (sketch). We simulate each transition t by the sequential firing of several transitions satisfying the requirement above. Assume $Var(t) = \{x_1, ..., x_n\}$. We add transitions $t_1, ..., t_n$ so that t_i is used to remove and add the tokens to which x_i is instantiated (using each of them a single variable x for that purpose). In order to guarantee that they are fired sequentially, we add auxiliary places, controlled by arcs labeled by ϵ. One of these auxiliary places also guarantees that the simulation of a transition is done atomically, that is, whenever such a simulation is started, no other simulation can start until the former has finished. Notice that this simulation can introduce deadlocks (for instance, when we fire t_1 but we cannot continue with t_2 due to abscess of tokens), but it does preserve reachability. Fig. 2 illustrates the previous construction when $Var(t) = \{x, y\}$.

Corollary 1. *Reachability of* $\emptyset$ *is undecidable for asynchronous* ν*-PN.*

4 Resource-constrained workflow nets

We propose here a presentation of rcwf-nets slightly different from the presentation of [3], though equivalent. We directly define rcwf-nets using (asynchronous) ν-PN, in order to shorten the gap between rcwf-nets and ν-PN.

Given a ν-PN $N = (P, T, F)$ and $x \in Var$ we define the P/T net $N_x = (P, T, F_x)$, where $F_x(n, m) = F(n, m)(x)$. Moreover, for $Q \subseteq P$, by $F|_Q$ we mean F restricted to $(Q \times T) \cup (T \times Q)$. We are now ready to define rcwf-nets.

Definition 2. *A* resource constrained wf-net *(or rcwf-net) is an asynchronous ν-PN $N = (P, T, F)$ such that:*

- *for all $t \in T$, $\nu \notin Var(t)$,*
- *$N_p = (P_D, T, F|_{P_D})_x$ is a wf-net.*

N_p is the P/T net obtained by removing static places, which we call *production net* of N. Then, a rcwf-net is an asynchronous ν-PN that does not create new tokens (because the variable ν does not label any arc) and such that its production net is a wf-net. In particular, it contains two special places *in* and *out* given by the definition of wf-nets. When there is no confusion we will simple refer to these places as *in* and *out*, respectively.

Definition 3. *Let $N = (P, T, F)$ be a rcwf-net and $m_0 \in P_S^{\oplus}$. For any $k \geq 0$, we define m_0^k, as the marking of N given by:*

- *$m_0^k(s)$ contains $m_0(s)$ black tokens, for each $s \in P_S$,*
- *$m_0^k(in)$ contains k pairwise different names,*
- *$m_0^k(d)$ is empty for every $d \in P_D \setminus \{in\}$.*

Moreover, for m_0^k we define the set of final markings $\mathcal{M}_{out}^k$ that contain the same k names in out, and empty in the rest of the dynamic places.

Notice that in the final markings we are not fixing the amount of tokens in static places, unlike in [3, 4].

Definition 4. *Let $N = (P, T, F)$ be a rcwf-net and $m_0 \in P_S^{\oplus}$. We say N is dynamically sound for m_0 if for each $k \geq 0$ and for each m reachable from m_0^k, we can reach some marking in $\mathcal{M}_{out}^k$.*

5 Undecidability of dynamic soundness

In this section we prove undecidability of dynamic soundness for rcwf-nets by reducing reachability for asynchronous ν-PN, which is undecidable, to it.

For this purpose, given an asynchronous ν-PN N, an initial marking m_0 of N (which we can assume to contain a single token in a given place i), we are going to construct a rcwf-net N' which is dynamic sound if and only if the empty marking is not reachable from m_0. Intuitively, the runs of N' will be divided into four steps: In the first step, the net gets ready for the simulation; in the second step, the initial marking m_0 of N is set; the third step simulates N; and finally, the last step is intuitively used to force that $\emptyset$ is not reachable if and only if N' is dynamically sound.

Let us explain with detail the four steps. In order to control in which step we are in, we consider four static places *step*1, *step*2, *step*3 and *step*4, that will be marked in mutual exclusion. Initially, *step*1 is marked.

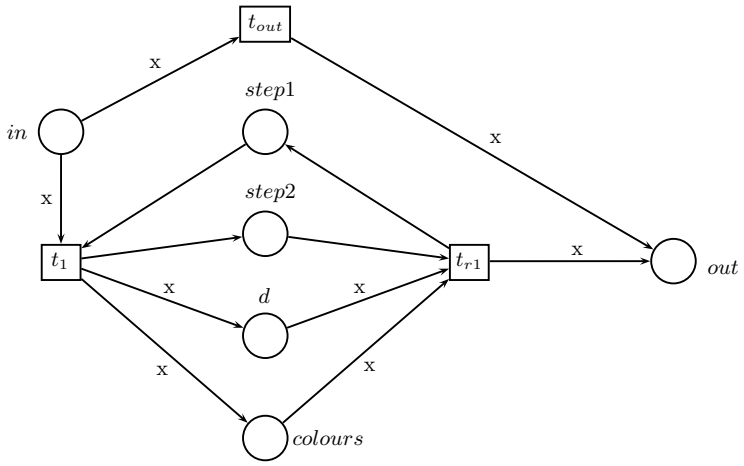

Fig. 3. Step 1

5.1 Step 1: Getting ready

First of all, as we want to build a rcwf-net, we add two special places *in* and *out*. We add a transition t_{out} which can move a token from *in* to *out*. This transition does not have any other precondition, so that it can be fired in any of the steps.

We will also consider two dynamic places, d and *colours*. The purpose of d will be explained in the last step. The place *colours* will store all the colours that we will use in the simulation of N, so that each transition in the construction which takes a token from *in*, will add it to *colours*. We store all the colours in order to be able to add them to *out* even if N consume all the tokens of some color. We need the place *colours* because N could erase some names, but we cannot do this in N' without being dynamically unsound.

In this first step, a transition t_1 is fired, removing a token from *in* and adding it to the two dynamic places d and *colours*. The purpose of placing a token in d will be explained later, in the last step. It also moves the toke from *step*1 to *step*2, thus moving on to the next step.

Finally, we need the firing of t_1 to be "reversible" (for the case in which we have a single name in *in*). Therefore, we add a new transition t_{r1} which moves a token from *step*2 to *step*1, removes the tokens in *colours* and d, and adds a token of the same color to *out* (not to *in*, since it cannot have incoming arcs). Fig. 3 illustrates the first step.

5.2 Step 2: Setting the initial marking

In order to simulate the behavior of N, we consider in N' the set of places of N. In this step we set the initial marking, which consists only of a name in the place of N that we call i. Therefore, we take a token from *in* and put it both in i and in *colours*. Moreover, we move the token from *step*2 to *step*3.

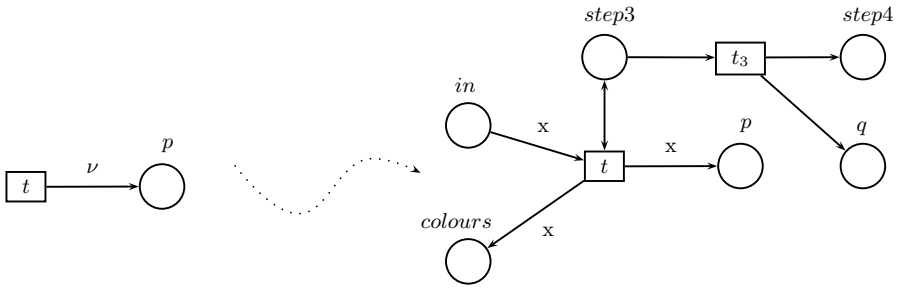

Fig. 4. Step 3

5.3 Step 3: Simulating N

In this step we simulate the behavior of N. Since N is an asynchronous ν-PN, it only uses variables x, ν and ϵ. Since N' is a rcwf-net, we have to simulate the creation of new names without using ν. We do it analogously as in the previous steps, by taking from in a name whenever one must be created, and placing it both in $colours$ and whatever places pointed by arcs labeled by ν. Since all the names contained in the place in are different, this is a correct simulation of the creation of a fresh name.

It may be the case that at some point there are no more tokens in the place in, so that no more name creations can be simulated. Therefore, a run of N' with k different names in the place in simulates a run of N in which at most k names are used (actually, $k-1$ because of the name that is put in d). Notice that the dynamic soundness has to do with the behavior of a rcwf-net from any initial marking, so that all the behaviors of N will be considered.

In this step we add $step3$ both as precondition and postcondition of any transition in N, so that transitions in N can only be fired in this step. At any point, we can fire a transition t_3 that moves the token from $step3$ to $step4$, thus finishing the simulation of N. Moreover, it also puts a black token in a new static place q, whose purpose we will explain later. Figure 4 shows the simulation of a transition with a ν.

5.4 Step 4: Reducing reachability to dynamic soundness

When the fourth step starts, there is a name in d, a black token in $step4$ (which will stay there until the end of the execution of N') and in q, the set of names that have been used along the execution of the rcwf-net is stored in $colours$ and the places of N are marked with a marking which is reachable in N.

We add a transition t_f, which can move all the tokens from $colours$ to out, and with $step4$ both as precondition and postcondition, so that it cannot be fired until this step starts.

We want to force N' to be dynamically unsound whenever $\emptyset$ is reachable. Since we can move names directly from in to out, we need to build a marking from which it is not possible to remove names from places different from out.

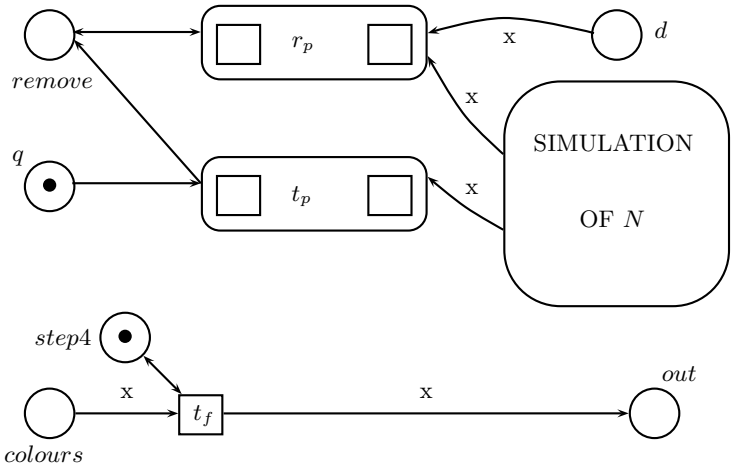

Fig. 5. Step4

We add to N' a transition t_p for each place p of N. When q is marked, there is a choice between all the transitions t_p, each of which removes a token from p, and puts a black token in a static place *remove*. Intuitively, we are only able to fire some t_p if the current marking of N is not $\emptyset$. Otherwise, if t_3 was fired exactly from $\emptyset$, then no transition t_p can be fired.

If we are able to fire some t_p then we have a token in *remove*. In that case, we can fire transitions r_p for each dynamic place p (different from *colours*, *in* and *out*), that removes a token from p, and puts the token back to *remove*. Therefore, if *remove* is marked, we can empty every dynamic place different from *colours*, *in* and *out*. In particular, the firing of r_d is the only way to remove the token in d. Figure 5 sketches how the fourth step is performed.

5.5 Undecidability

Now we are ready to prove that the previous construction reduces reachability for asynchronous ν-PN to dynamic soundness for rcwf-nets.

Proposition 2. *Given a ν-PN N with initial marking m_0, the rcwf-net N' built is dynamically sound if and only if $\emptyset$ is not reachable from m_0 in N.*

Proof. First, let us suppose that $\emptyset$ is reachable from m_0 in N. Let n be the number of different names created in some run that reaches $\emptyset$. If we consider the net N' with $n+1$ or more instances (that is, with at least $n+1$ different names in the place *in*), then we can reach a marking m' of N' in which the places of N are unmarked, the names that have been used in the computation are stored in *colours*, d is marked by a color and *step*4 and q are marked with black tokens. From this marking, we cannot fire any of the t_p transitions, and therefore, we cannot remove the token from q. Therefore, *remove* cannot be marked, which

is the only way in which the name in d can be removed. Summing up, from the initial marking with $n+1$ different names in in we have reached a marking from which we cannot reach a final marking of N' (that in which the only marked dynamic place is out), so that N' is not dynamically sound.

Conversely, let us suppose that $\emptyset$ is not reachable. We have to prove that for each $k \geq 0$ and for each m reachable from m_0^k, we can reach some marking in $\mathcal{M}_{out}^k$. Let us consider several cases, depending on which step the considered marking is in.

- If $step1$ is marked in m then all the names are either in the place in or in out. Therefore, we can fire t_{out} repeatedly, transferring all the tokens in in to out, and we are done.
- If $step2$ is marked in m we can fire t_{r1}, reaching a marking in which $step1$ is marked, so we can apply the previous case.
- If $step3$ is marked in m we can fire t_3, reaching a marking in which $step4$ is marked. We discuss this case next.
- If $step4$ is marked in m we can fire t_f repeatedly, putting all the names that have been used by the construction in out, thus emptying $colours$. Moreover, we can fire t_{out} repeatedly, moving all the tokens which remain in in to out. Therefore, all the tokens that initially were set in in, are set in out, so we only have to prove that we can empty the other dynamic places. If $step4$ is marked then there must be a token in q or $remove$. If the token is in q, since $\emptyset$ is not reachable, there is some name in some place p of N. Therefore, we can fire the transition t_p from m, reaching a marking in which $remove$ is marked. Finally, if $remove$ is marked in m, we can remove all the tokens from the dynamic places different from $colours$, in and out, reaching the desired marking.

The previous result proves that reachability of the empty marking in asynchronous ν-PN, which is undecidable, can be reduced to dynamic soundness for rcwf-nets. Therefore, we finally obtain the following result:

Corollary 2. *Dynamic soundness is undecidable for rcwf-nets.*

6 Decidability of dynamic soundness for a subclass of rcwf-nets

We have proved that dynamic soundness is undecidable in general. However, if we consider more restrictive requirements for our rcwf-nets, dynamic soundness turns decidable. In the literature, several notions of rcwf-nets and soundness have been studied, most of them being more restrictive than our general definition. In particular, in [3] the authors consider wf-nets which satisfy the following condition, which we have not required: for each node n, there are paths from in to n and from n to out. We are going to consider a less restrictive requirement, namely that every transition has some dynamic postcondition. In that case, and considering some very reasonable requirements, dynamic soundness is decidable

even if shared resources can be consumed or created by instances. This reasonable requirement is the following: when a single instance is given arbitrarily many global resources, then it evolves properly. This is equivalent to just removing static places.

Let $N = (P, T, F)$ be a wf-net. We denote by m_{in} the marking of N given by $m_{in}(in) = 1$ and $m_{in}(p) = 0$ for $p \neq in$. Analogously, we define m_{out} as the marking of N given by $m_{out}(out) = 1$ and $m_{out}(p) = 0$ for $p \neq out$. A wf-net N is *sound* [2] if for every marking m reachable from m_{in}, m_{out} is reachable from m. We are now ready to define our subclass of rcwf-nets:

Definition 5. *We say that a rcwf-net $N = (P, T, F)$ is a* proper *rcwf-net if the two following conditions hold:*

- *for each $t \in T$, $t^\bullet \cap P_D \neq \emptyset$,*
- *the production net N_p of N is sound.*

Intuitively, the behavior of N_p represents the maximal behavior of each instance of N. In particular, if m is a reachable marking of a rcwf-net N, then the markings of N_p obtained by projecting m to each of the names in m are all reachable too.

In [3, 4] other classes more restricted than proper rcwf-nets are defined.[1] However, the previous conditions are enough for our decidability result, and indeed our requirement can be deduced from the conditions required in [3, 4].

Lemma 1. *The production net N_p of a proper rcwf-net N is bounded.*

Proof. Let us suppose that N_p is sound and unbounded (assuming the initial marking m_0^1). Then, there are markings of N_p, m_1, m_2, and m_1' such that $m_0^1 \rightarrow^* m_1 \rightarrow^* m_2 = m_1 + m_1'$ with m_1' non empty. Since N_p is sound, $m_1 \rightarrow out$, so that $m_2 = m_1 + m_1' \rightarrow^* out + m_1'$. Again, by soundness of N_p, it must be the case that $out + m_1' \rightarrow^* out$. Since $out^\bullet = \emptyset$, it must be the case that $m_1' \rightarrow^* \emptyset$, but this is not possible because N is proper (and, in particular, all the transitions of N_p have postconditions).

Actually, in the proof of decidability of dynamic soundness for proper rcwf-nets, we only need that the production net is bounded (and boundedness is decidable for P/T nets). By the previous result, we know that the production net of a proper rcwf-net is bounded, but even if our rcwf-net is not proper, we can still check whether its production net is bounded, in which case our proof still holds. We reduce dynamic soundness to a home space problem in P/T nets.

Let us explain intuitively how the construction works. It is similar to a construction used in [4]. Given a proper rcwf-net N, we know that N_p is bounded. Then, we can consider the state machine associated to the reachability graph of N_p. More precisely, if m is a reachable marking in N_p, then we will consider a place also denoted by m. A token in m stands for an instance of N in state m.

[1] E.g., by demanding that there are paths from *in* to every node, and from every node to *out*.

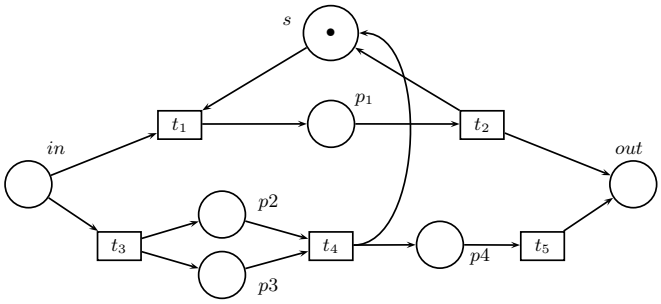

Fig. 6. A proper rcwf-net N

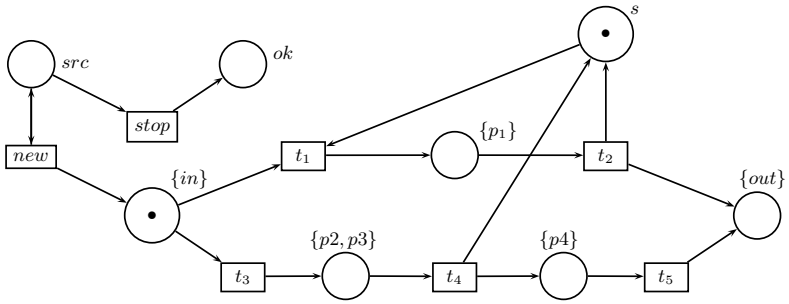

Fig. 7. N^{tr} obtained by applying Def. 6 to N in Fig. 6 (omitting the arcs from ok)

Notice that this is correct because all the markings reachable in N must be reachable in N_p (after projecting). So far, it is like in [4]. Moreover, the static places will be considered as places of the new net too, and will be pre/postconditions of the transitions we add, in the same way as they were in the original net.

Finally, we have to consider one more place src in order to set the initial number of instances that we are going to consider for the net. Let us denote by $\mathcal{R}(N)$ the set of markings reachable in a wf-net net N from m_{in}. Now we are ready to define the construction which will let us prove the decidability of dynamic soundness.

Definition 6. *Let $N = (P, T, F)$ be a proper rcwf-net and $m_0^s \in P_S^{\oplus}$. We define the P/T net $N^{tr} = (P^{tr}, T^{tr}, F^{tr})$ as follows:*

- $P^{tr} = P_S \cup \mathcal{R}(N_p) \cup \{src, ok\}$,
- $T^{tr} = \{(m_1, t, m_2) \in \mathcal{R}(N_p) \times T \times \mathcal{R}(N_p) \mid m_1 \xrightarrow{t} m_2 \text{ in } N_p\} \cup \{new, stop\}$,
- F^{tr} *is such that:*
  - $F^{tr}(m_1, (m_1, t, m_2)) = F^{tr}((m_1, t, m_2), m_2) = 1$,
  - $F^{tr}(src, stop) = F^{tr}(stop, ok) = 1$,
  - $F^{tr}(src, new) = F^{tr}(new, src) = F^{tr}(new, in) = 1$,
  - $F^{tr}(ok, (m_1, t, m_2)) = F((m_1, t, m_2), ok) = 1$,
  - *If* $s \in P_S$, $F^{tr}((m_1, t, m_2), s) = F(t, s)$ *and* $F^{tr}(s, (m_1, t, m_2)) = F(s, t)$,
  - $F^{tr}(x, y) = 0$, *otherwise.*

The initial marking of N^{tr} *is* m_0^{tr}*, given by* $m_0^{tr}(src) = 1$*,* $m_0^{tr}(m) = 0$ *for* $m \in \mathcal{R}(N_p)$ *and* $m_0^{tr}(s) = m_0^s(s)$ *for* $s \in P_S$*.*

Figure 7 shows the previous construction for the net in Fig. 6. Note that N^{tr} is finite because N_p is bounded, so that it can be effectively computed. Intuitively, N^{tr} creates by means of transition *new* several instances in its initial state, after which if fires *stop*, marking place *ok*, which is a precondition of the rest of the transitions, so that from then on they can be fired.[2] Each token in a place $m \in \mathcal{R}(N_p)$ of N^{tr} represents an instance of N, running concurrently with other instances and sharing the resources in the static places with them. Therefore, the net will simulate runs of as many instances of the original net as times the transition *new* has been fired. Let us define a correspondence between the markings of N and the markings of N^{tr}.

Definition 7. *Given a marking* m *of* N*, we define the marking* m^{tr} *of* N^{tr} *as follows:* $m^{tr}(src) = 0$*,* $m^{tr}(ok) = 1$*,* $m^{tr}(s) = m(s)(\bullet)$ *for* $s \in P_S$*, and* $m^{tr}(m') = |\{a \in Id(m) \mid m(p)(a) = m'(p)\ \forall p \in P_D\}|$*, that is, the number of instances in state* m'*.*

Notice that all the markings reachable in N^{tr} with *ok* marked are of the form m^{tr} for some marking m reachable in N. The following result is trivial by construction of N^{tr}.

Lemma 2. $m_0^k \rightarrow^* m$ *in* N *if and only if* $m_0^{tr} \stackrel{new^k \cdot stop}{\longrightarrow} (m_0^k)^{tr} \rightarrow^* m^{tr}$*. Moreover, all the computations in* N^{tr} *start by firing new* $k \geq 0$ *times, possibly followed by stop, in which case* $(m_0^k)^{tr}$ *is reached.*

Finally, we are ready to prove that this construction reduces the dynamic soundness problem for proper rcwf-nets to the home space problem for P/T nets. We denote by e_p the marking given by $e_p(p) = 1$ and $e_p(q) = 0$ for $p \neq q$.

Proposition 3. *Let* N *be a proper rcwf-net.* N *is dynamically sound if and only if the linear set* $\mathcal{L}$ *generated by* $\{out\} \cup \{e_s \mid s \in P_S\}$ *is a home space for* N^{tr}*.*

Proof. We start by remarking that $\mathcal{L}$ contains markings with any number of tokens in *out* and in static places, and empty elsewhere. Notice also that each transition different from *new* and *stop* has exactly one precondition in $\mathcal{R}(N_p)$ and one postcondition in $\mathcal{R}(N_p)$. Therefore, after the firing of *stop*, the total number of tokens in places in $\mathcal{R}(N_p)$ remains constant. Therefore, if *new* is fired k times and a marking in $\mathcal{L}$ is reached, then necessarily this marking has k tokens in *out*. Finally, notice that $m \in \mathcal{M}_{out}^k$ iff $m^{tr} \in \mathcal{L}$ and it contains exactly k tokens in *out*.

Let us first suppose that N is not dynamically sound. Then, there is a $k > 0$ and a marking m reachable from m_0^k from which no marking in $\mathcal{M}_{out}^k$ is reachable. By Lemma 2, the marking m^{tr} is reachable after firing *new* k times. Then, from

[2] Actually, the construction still works without place *ok*, though it simplifies the forthcoming explanations.

m^{tr} no marking in $\mathcal{L}$ can be reached. Indeed, if some marking m'^{tr} in $\mathcal{L}$ is reached from m^{tr} it has necessarily k tokens in *out* and again by Lemma 2, $m' \in \mathcal{M}^k_{out}$ is reached in N, contradicting our first hypothesis. Then, $\mathcal{L}$ is not a home space and we conclude this implication.

Reciprocally, let us assume that $\mathcal{L}$ is not a home space. Then, there is a reachable marking of N^{tr} from which no marking of $\mathcal{L}$ can be reached. Let us suppose that this marking is of the form m^{tr} (otherwise, we consider the marking obtained after firing *stop*, and no marking of $\mathcal{L}$ can be reached from it). Let us suppose that there are k tokens in places in $\mathcal{R}(N_p)$ in m^{tr}. Then, by Lemma 2 and the previous remarks (analogously to the previous case) no marking in $\mathcal{M}^k_{out}$ can be reached from m, so that N is not dynamically sound.

Finally, as the home space problem is decidable for linear sets of markings of P/T nets [9], we obtain the following result:

Corollary 3. *Dynamic soundness for proper rcwf-nets is decidable.*

7 Conclusions and future work

In this paper we have continued the study of concurrent workflow processes that share some global resources, first studied in [3] and more recently in [4]. In particular, we consider resource-constrained workflow nets in which each instance is allowed to consume or create new resources.

We have first established the undecidability of dynamic soundness for rcwf-nets when the use of resources is unrestricted, so that each instance is allowed to terminate a run having consumed or created global resources. Such result is achieved by means of an alternative presentation of rcwf-nets which is closer to ν-PN. More precisely, we have defined a subclass of ν-PN in which processes can only interact asynchronously with each other via a global shared memory, thus bringing together ν-PN and rcwf-nets. We have then seen that the undecidability of the reachability problem for ν-PN can be transfered to its asynchronous subclass. Although we have focused on reachability, we claim that most undecidability results can also be transfered, so that both classes are essentially equivalent. Then we have reduced this reachability problem to dynamic soundness of rcwf-nets. This reduction is not at all trivial, even though the alternative presentation of rcwf-nets eases the simulation of ν-PN by means of them (third step of the reduction).

Then we have considered a subproblem of the latter. In the first place, we assume that each instance is sound when it is given infinitely many resources (which amounts to saying that its behavior is not restricted by global resources). Moreover, we assume a technical condition, which is weaker than the standard "path property", that intuitively meas that all transitions are significant in the completion of the task. Under these hypotheses, we prove that dynamic soundness is decidable, by reducing it to a home space problem for a linear set of home markings, which is decidable.

There are many ways in which this work must be continued. The most important one could be to bridge the gap between our undecidability and our decidability result. In other words, we must address the problem of dynamic soundness whenever the path property holds, without assuming that a single instance of the net is necessarily sound (and always without assuming that instances give back the resources they use). Notice that our undecidability proof is no longer valid if we assume that property (indeed, the step 4 of our construction has transitions without postconditions), and it does not seem possible to easily fix this issue.

A dynamically sound rcwf-nets in which some proper runs may consume global resources without returning them must necessarily have other runs in which there is no need to consume global resources. Intuitively, the first run may be more desirable than the second one in terms of some measure which lies outside of the model. In this sense, a *priced* extension of rcwf-nets [12] in which runs from *in* to *out* have an associated cost could be interesting to be studied.

References

1. van der Aalst, W.M.P., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT Press (2002)
2. van der Aalst, W.M.P.: Verification of workflow nets. In Azéma, P., Balbo, G., eds.: ICATPN. Volume 1248 of Lecture Notes in Computer Science., Springer (1997) 407–426
3. van Hee, K.M., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Soundness of resource-constrained workflow nets. In Ciardo, G., Darondeau, P., eds.: ICATPN. Volume 3536 of Lecture Notes in Computer Science., Springer (2005) 250–267
4. Juhás, G., Kazlov, I., Juhásová, A.: Instance deadlock: A mystery behind frozen programs. In Lilius, J., Penczek, W., eds.: Petri Nets. Volume 6128 of Lecture Notes in Computer Science., Springer (2010) 1–17
5. van der Aalst, W.M.P., van Hee, K.M., ter Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of workflow nets with reset arcs. T. Petri Nets and Other Models of Concurrency **3** (2009) 50–70
6. Rosa-Velardo, F., de Frutos-Escrig, D.: Name creation vs. replication in petri net systems. Fundam. Inform. **88** (2008) 329–356
7. Rosa-Velardo, F., de Frutos-Escrig, D., Alonso, O.M.: On the expressiveness of mobile synchronizing petri nets. Electr. Notes Theor. Comput. Sci. **180** (2007) 77–94
8. Rosa-Velardo, F., de Frutos-Escrig, D.: Decision problems for petri nets with names. CoRR **abs/1011.3964** (2010)
9. de Frutos-Escrig, D., Johnen, C.: Decidability of home space property. Technical Report LRI-503, Univ. de Paris-Sud, Centre d'Orsay, Laboratoire de Recherche en Informatique (1989)
10. Esparza, J., Nielsen, M.: Decidability issues for petri nets - a survey. Bulletin of the EATCS **52** (1994) 244–262
11. Reisig, W.: Petri Nets: An Introduction. Volume 4 of Monographs in Theoretical Computer Science. An EATCS Series. Springer (1985)
12. Abdulla, P.A., Mayr, R.: Minimal cost reachability/coverability in priced timed petri nets. In de Alfaro, L., ed.: FOSSACS. Volume 5504 of Lecture Notes in Computer Science., Springer (2009) 348–363