Fundamenta Informaticae XX (2013) 1–26
IOS Press

Safety and soundness for Priced Resource-Constrained Workflow nets

María Martos-Salgado[*]
Facultad de Informática, Universidad Complutense de Madrid
mrmartos@estumail.ucm.es

Fernando Rosa-Velardo[*]
Facultad de Informática, Universidad Complutense de Madrid
fernandorosa@sip.ucm.es

Abstract. We extend workflow Petri nets (wf-nets) with discrete prices, by associating a price to the execution of a transition and to the storage of tokens. We first define the safety and the soundness problems for priced wf-nets. A priced wf-net is safe if no execution costs more than a given budget. The soundness problem is that of deciding whether the workflow can always terminate properly, where in the priced setting "properly" also means that the execution does not cost more than a given threshold. Then, we study safety and soundness of resource-constrained workflow nets (rcwf-nets), an extension of wf-nets for the modeling of concurrent executions of a workflow, sharing some global resources. We develop a framework in which to study safety and soundness for priced rcwf-nets, that is parametric on the cost model. Then, that framework is instantiated, obtaining the cases in which the sum, the maximum, the average and the discounted sum of the prices of all instances are considered. We study the decidability and the complexity of these properties, together with their relation.

1. Introduction

Workflow nets (wf-nets) are an important formalism for the modeling of business processes, or workflow management systems [2, 3]. Roughly, a wf-net is a Petri net with two special places, in and out. Its initial

[*]Authors supported by the Spanish projects STRONGSOFT TIN2012-39391-C04-04, DESAFIOS10 TIN2009-14599-C03-01 and PROMETIDOS S2009/TIC-1465.
Address for correspondence: Facultad de Informática. C$\backslash$Prof. José García Santesmases, s/n - 28040 Madrid (Spain)

marking is that with a token in the place in and empty everywhere else, which models the situation in which a task has been scheduled. The basic correctness notion for a workflow is that of soundness.

Intuitively, a workflow is sound if it cannot go wrong, so that the scheduled task can always be completed. This implies that no supervisor is needed in order to ensure its completion (under a fairness assumption that rules out the presence of livelocks). More precisely, and in terms of wf-nets, soundness implies that at any reachable state, it is possible to reach the final state, that with a token in the place out, and empty elsewhere. Soundness is decidable for wf-nets, and even polynomial for free-choice wf-nets [3].

Recent works [19, 20, 24] study an extension of wf-nets, called resource-constrained wf-nets (rcwf-nets) in which several instances of a workflow execute concurrently, assuming that those instances share some global resources. Even if a single instance of an rcwf-net is sound, several instances could deadlock because of these shared resources. In [19] the authors define dynamic soundness, the condition stating the existence of a minimum amount of resources for which any number of instances running simultaneously can always reach the final state, that in which all the tasks have been completed and the shared resources are returned. The paper [20] defines another notion of dynamic soundness, in terms of the absence of instance deadlocks in rcwf-nets, fixing the initial amount of resources though keeping the condition that instances must not change the number of resources. In [24] we continued the work in [20], but we considered that instances may create or consume resources. We proved this notion of dynamic soundness to be undecidable, and we identified a subclass of rcwf-nets, called proper, for which it is decidable.

In the fields of business process management or web services, the importance of QoS properties in general and cost estimation in particular has been identified as central in numerous works [29, 23, 30, 22, 26, 14, 15]. As an example, in the previously mentioned models, it may be possible to reach the final marking in different ways, due to the different interleavings of the execution, or to the inherent non-determinism in wf-nets. Moreover, in the case of rcwf-nets, an instance locking some resource may force another instance to take a "less convenient" path (in terms of money, energy or gas emissions, for example), that does not use the locked resource. However, the reason why a workflow should prefer one path over another is something that lies outside the model.

As a motivating example, let us consider the following real scenario, taken from the Chemistry Degree of the Complutense University, where every student of the second course must do a practical exercise which consists in synthesizing two different chemical components and comparing their properties. Each student has to perform several steps. For that purpose, they need to use some devices. We can model the procedure that students have to follow as a wf-net, and the interaction of all the students (with limited resources) as a rcwf-net. In most of the steps, the use of the corresponding device has a cost. Later, we will see that this kind of costs correspond to *firing costs*. However, these are not the only costs we need to take into account. When all the devices of a certain kind are being used at the same time, there may be students who need to wait to use these devices and the components they have made until this moment, may go wrong because of the delay if they do not keep them at some specific conditions, as a specific temperature, for example. Therefore, there are costs that depend on the number of students that are waiting to use the devices. These costs will correspond to *storage costs*, that is, costs produced when a transition is fired and there are some tokens in some specific places.

In order to model situations like this, in this paper we add prices to our nets, similarly as done for the (untimed) priced Petri nets in [5] or for causal nets in [14, 15]. We consider different types of prices, modeled as tuples of integers or naturals. More precisely, we add firing costs to transitions and storage costs to places. Then, the price of firing a transition is computed as the cost of its firing plus the cost of storing all the tokens in the net when the transition is fired. The price of a run is defined as the sum of the prices of the firings of its transitions. Then, we say that a workflow net is price-safe if the price of each of its runs stays under a given threshold. When costs are integers, we prove that price-safety is undecidable, so that in the rest of the paper we restrict ourselves to non-negative costs.

In this priced setting, we restate the soundness problems. For ordinary wf-nets, this is straightforward: a priced wf-net is sound if we can always reach the final marking without spending more than a given budget. We prove that price-soundness is decidable for wf-nets, even with negative costs.

For priced rcwf-nets, the definitions of safety and soundness are not so straightforward, since they must consider the behavior of an arbitrary number of instances. We consider parametric definitions: For any run, we collect the prices of every instance in the run, so that safety and soundness are parametric in the way in which local prices are aggregated to obtain a global price. The definition is open to many different variants. We study several such variants in this paper: the maximum (denoted by Max), in which we consider the price of the most expensive instance; the sum of the prices of all the instances(Sum); the average of the prices of all the instances (Av); and the discounted sum (Ds), in which we consider a weighted sum of the prices of the instances, so that the first instances are more important than the last ones. For instance, in the case of Av, for each execution we compute the average of the prices of all the instances participating. Then, safety with respect to Av guarantees that all the computed averages (in every possible execution) do not exceed a given bound.

Back to the example, the price for the university of a student performing the whole exercise is the sum of the costs of using each device and the costs that come from other students waiting for the devices he/she is using. Then, the university could be interested in setting a bound for:

- The total amount of money spent by all students in a class;
- The money spent by each student;
- The average of the amounts spent by each student.

This corresponds to the study of priced safety and soundness of the corresponding model for the different price predicates.

We prove decidability of price-safety for the sum, the maximum, and a finite version of discounted sum, relying on the decidability of coverability for a class of Petri nets with names, broadcasts and whole place operations, that can be seen as an unordered version of Data Nets with name creation [4, 21]. In these cases we have decidability of priced soundness within the proper subclass. As a consequence, we obtain the corresponding results for ordinary priced wf-nets (with non-negative costs). For the average, we reduce soundness to the unpriced case, so that it is decidable for proper rcwf-nets. However, decidability of price-safety for the average remains open. Finally, we prove some relations between the studied predicates, and give a preliminary result regarding complexity. More precisely, we prove that price-safety for Max, Sum and Ds has a non primitive recursive complexity.

In parallel to the works on wf-nets and rcwf-nets, there has recently been an increasing interest in the study of quantitative aspects of both finite and infinite state systems. E.g. in [10] the authors consider quantitative generalizations of classical languages, using weighted finite automata, that assign real numbers to words, instead of boolean values. They study different problems, which are defined in terms of how they assign a value to each run. In particular, they assign the maximum, limsup, liminf, average and discounted sum of the transition weights of the run.

Numerous works extend timed automata with prices [5, 7, 8]. E.g., the paper [5] defines a model of Timed Petri Nets with discrete prices. In such model, a price is associated to each run of the net. Then, the reachability (coverability) threshold-problem, that of being able to reach (cover) a given final marking with at most a given price, is studied. This study is extended to the continuous case in [6]. In our setting, we require the workflow to behave correctly in any case, without the need of a supervisor, which in the priced setting means that no run reaching the final marking costs too much, as opposed to the threshold problems, in which the existence of one good run is considered. In [14, 15], time and costs are added to causal nets, a subclass of Petri nets for the modelling of business processes, and a methodology for evaluating their performance is presented.

Quantitative aspects of reactive systems are studied as energy games e.g. in [9, 11, 17]. For example, in [17] games are played on finite weighted automata, studying the existence of infinite runs satisfying several properties over the accumulated weights, as ensuring that a resource is always available or does not exceed some bound.

Outline.

Section 2 gives some notations we use throughout the paper. Section 3 extends wf-nets with prices. In Section 4 we extend rcwf-nets and give some basic results. In Section 5 we study some specific price predicates, studying the decidability of safety and soundness for them, setting some complexity results and relating the different price predicates. Finally, in Section 6 we present our conclusions. This paper is a revised and extended version of [25].

2. Preliminaries

A quasi-order $\leq$ over a set A is a reflexive and transitive binary relation over A. Given a quasi-order $\leq$, write that $a < b$ if $a \leq b$ and $b \not\leq a$. Given $B \subseteq A$, we denote $B{\downarrow} = \{a \in A \mid \exists b \in B, a \leq b\}$ the downward closure of B and we say that B is downward-closed if $B{\downarrow} = B$. Analogously, we define $B{\uparrow}$, the upward closure of B and say B is upward closed if $B{\uparrow} = B$. We denote by $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$, the naturals completed with their limit ω. We write $v[i]$ to denote the i^{th} component of $v \in \mathbb{N}_\omega^k$ and $\mathbf{0} = (0, ..., 0)$. We denote by $\leq$ the component-wise order in any $\mathbb{N}_\omega^k$ or $\mathbb{Z}^k$, and by $<$ its strict version. A (finite) multiset m over a set A is a mapping $m : A \to \mathbb{N}$ with finite support, that is, such that $supp(m) = \{a \in A \mid m(a) > 0\}$ is finite. 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 $a \in A$. For a multiset m and $\lambda \in \mathbb{N}$, we take $(\lambda * m)(a) = \lambda * m(a)$. 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 $a \in A$, and $|m| = \sum_{a \in supp(m)} m(a)$. We use set notation for multisets when convenient, with repetitions to account for multiplicities greater than one. We write $\{a_1, ..., a_n\} \leq^\oplus \{b_1, ..., b_m\}$ if there is an injection $h : \{1, ..., n\} \to \{1, ..., m\}$ such that $a_i \leq b_{h(i)}$ for each $i \in \{1, ..., n\}$.

Petri Nets.

A Place/Transition (P/T) net 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.

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 $t^{\bullet}(p) = F(t,p)$, ${}^{\bullet}p(t) = F(t,p)$ and $p^{\bullet}(t) = F(p,t)$. A marking m enables a transition $t \in T$ if ${}^{\bullet}t \subseteq m$. In that case t can be fired, reaching the marking $m' = (m - {}^{\bullet}t) + t^{\bullet}$, and we write $m \xrightarrow{t} m'$. A run r is a sequence $m_0 \xrightarrow{t_1} m_1 \xrightarrow{t_2} ... \xrightarrow{t_n} m_n$. If r_1 and r_2 are two runs so that r_1 finishes at the marking in which r_2 starts, we denote by $r_1 \cdot r_2$ the run starting with the transitions in r_1, followed by those in r_2, as expected.

Workflow Petri Nets.

A workflow Petri net [3] (shortly a wf-net) is a P/T net $N = (P, T, F)$ such that:

1. there are $in, out \in P$ with ${}^{\bullet}in = \emptyset$, $in^{\bullet} \neq \emptyset$, ${}^{\bullet}out \neq \emptyset$ and $out^{\bullet} = \emptyset$,
2. for each $n \in P \cup T$ there is a path $in = n_1, n_2, \ldots, n_j = n, \ldots, n_k = out$ with $F(n_i, n_{i+1}) > 0$ for $1 \leq i \leq k-1$.

We will call the second condition the *path property*. Abusing the terminology, we will sometimes say that a wf-net does not satisfy the path property, meaning that it only satisfies the first condition in the previous definition. When there is no confusion we will simply refer to the special places given by the previous definition as in and out, respectively. We denote by m_{in} the marking of N with a single token in in, and empty elsewhere. Analogously, m_{out} is the marking of N with a single token in out and empty elsewhere. There are several definitions of soundness of wf-nets in the literature. We will use one called weak soundness in [3]. A wf-net is *weakly sound* if for any marking reachable from m_{in} it is possible to reach m_{out}.

Petri Nets with dynamic name creation and whole place operations.

In order to model different instances running in the same net, we will use names, each name representing a different instance. A ν-PN is an extension of Petri nets in which tokens are names and fresh name creation can be performed. We define them here as a subclass of $w\nu$-PNs, a class of nets which we will need in Sect. 5.1, that also allows whole-place operations and broadcasts, similar to Data Nets [21]. Data Nets extend P/T nets by considering a linearly ordered and dense domain of tokens, and in which whole place operations can be performed. Therefore, $w\nu$-PNs can be seen as an unordered version of Data nets [21] in which names can be created fresh. When a transition t of a $w\nu$-PN is fired, four operations are performed: the subtraction of several tokens of different colors, whole-place operations (affecting every color in the same way), the creation of new names and the addition of tokens.

Let us consider a set Var of variables and $\Upsilon \subset Var$ a set of name creation variables. A $w\nu$-PN is a tuple $N = (P, T, F, G, H)$ where P and T are finite disjoint sets of places and transitions, respectively; for each $t \in T$, $F_t : P \to (Var \backslash \Upsilon)^{\oplus}$ is its subtraction function, $G_t : P \times P \to \mathbb{N}$ is its whole-place operations matrix, and $H_t : P \to Var^{\oplus}$ is its addition function. Moreover, if $x \in H_t(p) \setminus \Upsilon$ then $x \in F_t(p')$ for some $p' \in P$.

Let Id be an infinite set of names. A marking is any $m : P \to Id^{\oplus}$. An a-token in p is an occurrence of $a \in m(p)$. $Id(m)$ is the set of names appearing in m, that is, $Id(m) = \bigcup_{p \in P} supp(m(p))$. We denote by $Var(t) = \{x \in Var \mid \exists p \in P,\ x \in F_t(p) \cup H_t(p)\}$ and $Var(p) = \{x \in Var \mid \exists t \in T,\ x \in F_t(p) \cup H_t(p)\}$. A mode is a mapping $\sigma : Var(t) \to Id$ extended pointwise to $\sigma : Var(t)^{\oplus} \to Id^{\oplus}$. A

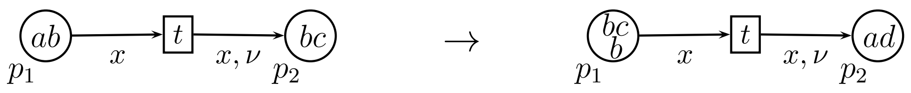

Figure 1. The firing of a $w\nu$-PN

transition t is enabled at a marking m with mode σ if for all $p \in P$, $\sigma(F_t(p)) \subseteq m(p)$ and for all $\nu \in \Upsilon$, $\sigma(\nu) \notin Id(m)$. Then, we say that t can be fired, reaching a new marking m', where for all $p \in P$, $m'(p) = \sum_{p' \in P}((m(p') - \sigma(F_t(p'))) * G_t(p', p)) + \sigma(H_t(p))$, which is denoted by $m \overset{t(\sigma)}{\rightarrow} m'$.

Example 2.1. Let $N = (\{p_1, p_2\}, \{t\}, F, G, H)$ be a $w\nu$-PN, where:

- $F_t(p_1) = \{x\}$, $F_t(p_2) = \emptyset$.
- $H_t(p_1) = \emptyset$, $H_t(p_2) = \{x, \nu\}$.
- $G_t(p_1, p_1) = 1$, $G_t(p_1, p_2) = 0$, $G_t(p_2, p_1) = 1$, $G_t(p_2, p_2) = 0$.

This net is depicted in Fig 1. Note that although F_t and H_t are represented by arrows labeled by the corresponding variables, the effects of G_t are not depicted. Let m be the marking of N such that $m(p_1) = \{a, b\}$ and $m(p_2) = \{b, c\}$. Then, t can be fired at m with mode σ, where $\sigma(x) = a$ and $\sigma(\nu) = d$, reaching a new marking m', such that $m'(p_1) = \{b, b, c\}$ and $m'(p_2) = \{a, d\}$. Note that m' is obtained from m by the following steps:

- Removing an a-token from the place p_1, due to the effect of F.
- Removing all tokens from p_2 and copying them to p_1, because of G.
- Adding an a-token and a d-token to p_2, because of H.

We write $m_1 \sqsubseteq m_2$ if there is a renaming m'_1 of m_1 such that $m'_1(p) \subseteq m_2(p)$ for every $p \in P$. A marking m is coverable from an initial marking m_0 if we can reach m' from m_0 such that $m \sqsubseteq m'$.

A $w\nu$-PN could be considered as an unordered Data Net, except for the fact that $w\nu$-PNs can create fresh names. In [4] the authors extend Data Nets with fresh name creation and prove that coverability is still decidable by instantiating the framework of Well Structured Transition Systems [18].

Proposition 2.1. Coverability is decidable for wν-PN.

Finally, we define ν-PN [28], which is a fragment of wν-PN without whole-place operations. Formally, a ν-PN is a $w\nu$-PN in which, for each $t \in T$, G_t is the identity matrix, and we will simply write (P, T, F, H).

In the rest of the paper we will introduce some more models, that will most of the times be priced versions of the models already defined. For the sake of readability, we prefer to present these models in an incremental way, instead of considering a general model which subsumes all the others.

3. Priced Workflow-nets

Let us define a priced extension of wf-nets. We follow the cost model in [5]. It essentially amounts to adding to a wf-net two functions, defining the price of the firing of each transition, and the cost of storing tokens when a transition is fired, respectively.

Definition 3.1. (Priced workflow net)
A *priced workflow net* (pwf-net) with *price arity* $k \geq 0$ is a tuple $N = (P, T, F, C, S)$ such that:

- (P, T, F) is a wf-net, called the underlying wf-net of N,
- $C : T \to \mathbb{Z}^k$ is a function assigning firing costs to transitions, and
- $S : P \times T \to \mathbb{Z}^k$ is a function assigning storage costs to pairs of places and transitions.

Notice that costs may be negative. The behavior of a pwf-net is given by its underlying wf-net. In particular, adding prices to a wf-net does not change its behavior, as the costs are not a precondition for any transition. That is the main difference between adding resources and prices. Indeed, firing costs could be seen as resources. However, since storage costs depend not only on the transitions which are fired, but also on the number of tokens in the rest of the places when the transitions are fired, they cannot be seen as resources anymore.

Let us now define the price of firing a transition and the price of a run.

Definition 3.2. (Price of a run)
Let t be a transition of a pwf-net enabled at a marking m. We define $\mathcal{P}(t, m)$, the *price of the firing of* t *at* m, as

$$\mathcal{P}(t, m) = C(t) + \sum_{p \in m - {}^\bullet t} S(p, t)$$

Then, the *price of a run* $r = m_1 \xrightarrow{t_1} m_2 \xrightarrow{t_2} m_3 \ldots m_n \xrightarrow{t_n} m_{n+1}$ of a pwf-net is $\mathcal{P}(r) = \sum_{i=1}^{n} \mathcal{P}(t_i, m_i)$.

Notice that in the definition of $\mathcal{P}(t, m)$ the term $m - {}^\bullet t$ is a multiset, so that if a place p appears twice in it then we are adding $S(p, t)$ twice in turn. It can be seen that firing costs can be simulated by storage costs, though we prefer to keep both to follow the approach in [5]. However, storage costs cannot be simulated by firing costs, since the former are marking dependent, while the latter are not. Next, we define safety of a pwf-net with respect prices.

Definition 3.3. (b-p-safety)
Given $b \in \mathbb{N}_\omega^k$, we say that a pwf-net is *b-p-safe* if for each run r reaching m_{out}, $\mathcal{P}(r) \leq b$.

Therefore, a pwf-net is b-p-safe if all the runs that reach the final marking cost less than the given budget. Next, we define soundness for pwf-nets.

Definition 3.4. (b-p-soundness)
Given $b \in \mathbb{N}_\omega^k$, we say that a pwf-net is *b-p-sound* if from each marking m, reachable from m_{in} via some run r_1, we can reach m_{out} via some run r_2 such that $\mathcal{P}(r_1 \cdot r_2) \leq b$.

Intuitively, for a pwf-net to be sound we need to be able to reach the final marking at any point with a price that does not exceed the budget $b \in \mathbb{N}^k_\omega$. It is easy to see that a pwf-net is b-p-sound iff it is weakly sound and b-p-safe. However, as we prove next, the latter is undecidable.

Proposition 3.1. b-p-safety is undecidable for pwf-nets.

Proof:
We reduce the cost-threshold-reachability problem for PPN with negative costs, and price arity 1, which is undecidable [5]. A PPN with arity 1 is a P/T net (P, T, F), endowed with a function $C : P \cup T \to \mathbb{Z}$ associating costs to transitions and places. Moreover, T is the disjoint union of T_0, the set of *instantaneous* transitions, and T_1, the set of *timed* transitions. The cost of firing $t \in T_0$ in any marking is just $C(t)$. The cost of $m \xrightarrow{t} m'$ with $t \in T_1$ is $C(t) + \sum_{p \in P} C(p) \cdot m(p)$. The cost of a run is the sum of all the transitions costs in it. The cost-threshold-reachability problem consists in, given m_f and $b \in \mathbb{N}$, decide whether there is a run σ with $m_0 \xrightarrow{\sigma} m_f$ such that $C(\sigma) \leq b$. It is proved in [5] that this problem is undecidable.

Given a PPN $N = (P, T, F, C)$, a final marking m_f and $b \in \mathbb{N}$, let us build a pwf-net N' as follows. First, we add new places, p_i, p_0, in and out, and transitions t_i, t_0 and t_f. Transition t_0 sets the initial marking of N, and t_f has m_f as precondition and puts a token in out. Places p_i and p_0 and transition t_i are added in order to make N' satisfy the second condition of the definition of workflow net. In order to satisfy the path property, at the beginning of each run we put a token in each place of the net, and remove them by firing t_i and t_0. Moreover, p_0 will be connected to each transition of the net and will be emptied in the final step, when t_f is fired. More precisely:

- Transition t_i takes a token from in, and puts a token in each place of N' except for p_0, in and out, connecting each place with in.
- We set each place of N' except for in, p_0 and out as a precondition of t_0, and p_0 and every place of m_0 as a postcondition of t_0, connecting each place except for out to p_0.
- We make p_0 be a precondition and postcondition of each $t \in T$, connecting each transition to p_0.
- Finally, we set p_0 and every place of m_f as a precondition of t_f, connecting each place and transition of the net with out.

The new places have no storage cost, t_0 has firing cost $b + 1$ and t_f and t_i have firing cost 0. For every $t \in T$ we take $-C(t)$ as the firing cost of t in N'. Moreover, if t is an instantaneous transition we set $S(p, t) = 0$ for every $p \in P$, and if it is a timed transition we take $S(p, t) = -C(p)$ for every $p \in P$.

By construction, if r is a run in N with cost c, then $t_i t_0 \cdot r$ is a run in N' with cost $b + 1 - c$. Let us see that there exists a run r of N such that $m_0 \xrightarrow{r} m_f$ with $C(r) \leq b$ iff N' is not 0-p-safe. Let r be a run such that $m_0 \xrightarrow{r} m_f$ and $C(r) \leq b$. Let us call $r' = t_i t_0 r t_f$ the corresponding run of N'. Then, $C(r') = 1 + b - C(r) \geq 1$. Therefore, N' is not 0-p-safe. Conversely, suppose that for every run r of N with $m_0 \xrightarrow{r} m_f$, $C(r) > b$. Then, for every run r' of N' reaching m_{out}, necessarily of the form $t_i t_0 r t_f$, $C(r') = 1 + b - C(r) \leq 0$. Therefore, N' is b-p-sound. □

Despite of the previous result, we can prove that b-p-soundness is decidable. We make use of the fact that weakly sound wf-nets are bounded, which can be proven following similar arguments to those in [1] for soundness.

Lemma 3.1. Let N be a wf-net. If N is weakly sound then N is bounded.

Proof:
Assume by contradiction that N is unbounded, so that for every $n \in \mathbb{N}$ there is a reachable marking m_n such that $|m_n| > n$. Since the reachability trees of wf-nets are finitary, König's lemma implies that we can assume those markings to be in the same run of N. Then, by Dickson's lemma [16], there are m and $m' \neq \emptyset$ such that $in \rightarrow^* m \rightarrow^* m + m'$. Since N is weakly sound, we have $m \rightarrow^* out$, so that $m + m' \rightarrow^* out + m'$. Again, because N is weakly sound $out + m' \rightarrow^* out$ must hold, so that $m' \rightarrow^* \emptyset$ (because $out^\bullet = \emptyset$). But this is not possible, since the path property implies that all the transitions of N have postconditions. We have reached a contradiction, so that N is necessarily bounded. □

Proposition 3.2. b-p-soundness is decidable for pwf-nets.

Proof:
Let N be a pwf-net. If the underlying wf-net N' of N is not weakly sound, then there is a reachable marking from which m_{out} is not reachable, and therefore N is not b-p-sound. Let us suppose that N' is weakly sound. Then, N' is bounded (previous lemma) and therefore we can build its reachability graph, which is finite. More precisely, we build the graph whose nodes are the reachable markings of N', and there is an arc connecting two markings m_1 and m_2 if and only if $m_1 \xrightarrow{t} m_2$ for some $t \in T$, labeled by $\mathcal{P}(t, m_1)$. In that way, the price of a run of N corresponds to the length of the corresponding path in the reachability graph. Since N' is weakly sound, N is not b-p-sound if and only if there is a run from m_{in} to m_{out} with a price which is not smaller than b, that is, if and only if there is a path from m_{in} to m_{out} in the reachability graph with a length c, and $i \in \{1, ..., k\}$, such that $c[i] > b[i]$. Therefore, in this case b-p-soundness can be reduced to computing the length of the longest path from m_{in} to m_{out} in the finite reachability graph (assumed to be infinite if no such path exists due to the presence of cycles with a positive length). This can be done polynomially in the number of states, even with negative weights [12]. □

In the following section we will consider a more general version of the problem, that in which several instances of the workflow execute concurrently, called resource-constrained workflow nets. Then, we will obtain other (positive) results regarding pwf-nets as a consequence of the more general problem.

To conclude this section, notice that if a pwf-net is b-p-safe (b-p-sound) then it is also b'-p-safe (b'-p-sound) for any $b' > b$, so that the set $\mathcal{B}(N) = \{b \in \mathbb{N}_\omega^k \mid N \text{ is } b\text{-p-safe } (b\text{-p-sound})\}$ is an upward-closed set. In this situation, we can apply the Valk & Jantzen theorem:

Theorem 3.1. ([31])
Let V be an upward-closed set. We can compute a finite basis of V if and only if for each $v \in \mathbb{N}_\omega^k$ we can decide whether $v\!\downarrow \cap V \neq \emptyset$.

Therefore, we can compute a finite basis of the set $\mathcal{B}(N)$, i.e., the minimal budgets b for which the pwf-net is b-p-safe (b-p-sound), provided we can decide b-p-safety (b-p-soundness) for each $b \in \mathbb{N}_\omega^k$.

4. Priced resource-constrained wf-nets

Let us start by recalling the definition of resource-constrained wf-nets (rcwf-nets). For more details see [24]. The definition we use is equivalent to those in [19, 20], though more convenient for our pur-

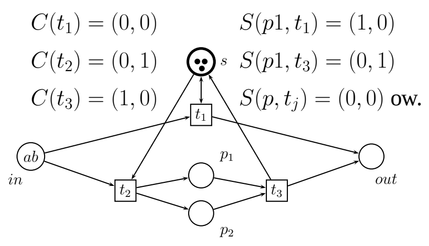

Figure 2. A priced resource-constrained workflow net

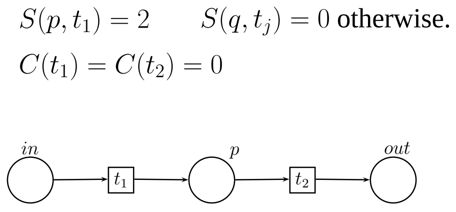

Figure 3. prcwf-net not $Sum(b)$ neither $Max(b)$-dynamically sound for any $b \in \mathbb{N}$

poses. We represent each instance in an rcwf-net by means of a different name. Hence, our definition of rcwf-nets is based on ν-PN. A ν-PN can be seen as a collection of P/T nets that can synchronize between them and be created dynamically [27]. We start by defining a subclass of ν-PN, called asynchronous ν-PN, in which each instance can only interact with a special instance (which models resources) that is represented by black tokens. We fix variables $\nu \in \Upsilon$ and $x, \epsilon \in Var \setminus \Upsilon$.

Definition 4.1. (Asynchronous ν-PN)
An *asynchronous ν-PN* is a ν-PN $N = (P, T, F, H)$ such that:

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

Places $p \in P$ with $Var(p) = \{\epsilon\}$ are called *static*, and are represented in figures by circles in bold. They can only contain (by construction) black tokens, so that they will represent resources, and can only be instantiated by ϵ. Places $p \in P$ with $Var(p) \subseteq \{x, \nu\}$ are called *dynamic*, and are represented by normal circles. They can only contain names (different from the black token), that represent instances, and can only be instantiated by x. We denote P_S and P_D the sets of static and dynamic places, respectively, so that $P = P_S \cup P_D$.

Let us introduce some notations we will need in the following definition. Given a ν-PN $N = (P, T, F, H)$ and $x \in Var$ we define the P/T net $N_x = (P, T, F_x)$, where $F_x(p,t) = F_t(p)(x)$ and $F_x(t,p) = H_t(p)(x)$ for each $p \in P$ and $t \in T$. Moreover, for $Q \subseteq P$, by $F|_Q$ we mean each F_t restricted to Q, and analogously for $H|_Q$. Roughly, an rcwf-net is an asynchronous ν-PN that does not create fresh names, and so that its underlying P/T net is a wf-net.

Definition 4.2. (Resource-constrained workflow nets)
A *resource-constrained workflow net* (rcwf-net) $N = (P, T, F, H)$ is an asynchronous ν-PN such that:

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

Fig. 2 shows an rcwf-net (for now, disregard the annotations C and S in the figure). In figures we do not label arcs, since they can be inferred (arcs to/from static places are labeled by ϵ, and arcs to/from dynamic places are labeled by x). N_p, the production net of N, is the P/T net obtained by projecting N to its dynamic places. Again, we will abuse terminology and say that a rcwf-net does not satisfy the path property, if its production net does not satisfy it.

Intuitively, each instance is given by a name, which is initially in in. Given $m_0 \in P_S^{\oplus}$, for each $j \in \mathbb{N}$ we define the initial marking m_0^j as the marking that contains $m_0(s)$ black tokens in each static place s, j pairwise different names in its place in, and is empty elsewhere. For instance, the marking of the rcwf-net in Fig. 2 is m_0^2, where $m_0 = \{s, s, s\}$. Moreover, for such m_0^j we denote by $\mathcal{M}_{out}^j$ the set of markings in which the same j names are in its place out and every other dynamic place is empty. Note that we do not impose any condition on the static places.

Now we define the subclass of proper rcwf-nets, which can be intuitively understood as the subclass of rcwf-nets behaving properly (being weakly sound) if endowed with infinitely-many resources, which amounts to removing the restriction of its behavior by means of resources.

Definition 4.3. (Proper rcwf-nets)
An rcwf-net is *proper* if its production net is weakly sound.

It is decidable to check that an rcwf-net is proper [3, 24]. As for wf-nets we have the concept of soundness,which is called *dynamic soundness* in this setting. Dynamic soundness corresponds to the idea that each instance running in the net can always finish correctly.

Definition 4.4. (Dynamic soundness)
We say that the prcwf-net N is *dynamically sound* for $m_0 \in P_S^{\oplus}$ if for each $j > 0$ and for each marking m reachable from m_0^j we can reach a marking $m_f \in \mathcal{M}_{out}^j$.

In [24] we proved that this property is decidable for proper rcwf-nets. Now we define the priced version of rcwf-nets, analogously as in Def. 3.1.

Definition 4.5. (Priced rcwf-net)
A *priced rcwf-net* (prcwf-net) with *price arity* $k \geq 0$ is a tuple $N = (P, T, F, H, C, S)$ such that:

- (P, T, F, H) is an rcwf-net, called the underlying rcwf-net of N,
- $C : T \to \mathbb{Z}^k$ and $S : P \times T \to \mathbb{Z}^k$ are functions specifying the firing and storage costs, respectively.

As for priced wf-nets, the behavior of a priced rcwf-net is given by its underlying rcwf-net. However, its runs have a price. Before we start giving the formal definitions of the price of a run, let us go back to the example in the introduction. The net in Fig. 4 is a very simple model of the chemistry exercise. We model each step as a transition, and tokens in places represent students who are ready to perform the next step. Performing steps two (transition t_2) and three (transition t_3) costs 1, so that $C(t_2) = C(t_3) = 1$. In the exercise, the step two can possibly go wrong, in which case the students need to repeat this step, which is possible thanks to t_1. The device used in those two steps is the same, and there is only one such device, which is modelled by the static place s with one token initially. Moreover, the cost of a student waiting in p_4 for the necessary device is 1, and therefore $S(p_4, t_2) = S(p_4, t_3) = 1$.

Let us now define the price of an instance in a run.

Definition 4.6. (Price of an instance)
We define the *price of an instance* $a \in Id(m_0)$ in a run $r = m_0 \xrightarrow{t_1(\sigma_1)} m_1 \xrightarrow{t_2(\sigma_2)} m_2 \ldots m_{n-1} \xrightarrow{t_n(\sigma_n)} m_n$ of a prcwf-net as

$$\mathcal{P}(a, r) = \sum_{\substack{i=1 \\ \sigma_i(x)=a}}^{n} \left(C(t_i) + \sum_{p \in P} |m_{i-1}(p) - \sigma(F_t(p))| * S(p, t_i) \right)$$

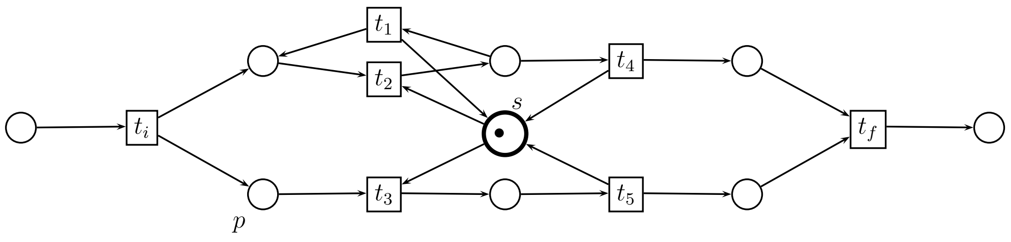

Figure 4. prcwf-net representing the practical exercise of the Chemistry degree

Intuitively, we are considering those transitions in r fired by a, and computing its price as we did in Def. 3.2 for pwf-nets. In particular, we are assuming that when computing the price of the firing of a transition by an instance, the tokens belonging to other instances are accounted for. In other words, a pays a penalization for the storage of all tokens when it fires a transition. We could have also decided that each instance only pays for its own tokens, thus being in a slightly different setting, but the techniques used in our results would also apply.

Since in rcwf-nets we are interested in the behavior of several concurrent instances, we collect their prices in the following definition.

Definition 4.7. (Price of a run)
Given a run r of a prcwf-net starting in m_0, we define the *price of* r as the multiset

$$\mathcal{P}(r) = \{\mathcal{P}(a, r) \mid a \in Id(m_0)\} \in (\mathbb{Z}^k)^{\oplus}.$$

Instead of fixing the condition to be satisfied by all the prices of each instance, we define a parametric version of p-safety and dynamic soundness. More precisely, those properties for prcwf-nets are parameterized with respect to a price-predicate.

Definition 4.8. (Price-predicate)
A *price-predicate* ϕ of arity $k \geq 0$ is a predicate over $\mathbb{N}^k_\omega \times (\mathbb{Z}^k)^{\oplus}$ such that if $b \leq b'$ and $A' \leq^{\oplus} A$ then $\phi(b, A) \to \phi(b', A')$ holds.

Intuitively, b stands for the budget, and A stands for the price of a run. Notice that price-predicates are upward-closed in their first argument, but downward-closed in their second argument. Intuitively, if a price-predicate holds for given budget and costs, then it holds with a greater budget and less costs, as expected. From now on, for a price-predicate ϕ and $b \in \mathbb{N}^k_\omega$, we will denote by $\phi(b)$ the predicate over $(\mathbb{Z}^k)^{\oplus}$ that results of specializing ϕ with b. Moreover, when there is no confusion we will simply say that a run r satisfies a predicate when $\mathcal{P}(r)$ satisfies it.

We now proceed as in the case of a single instance, defining p-safety and dynamic soundness, though with respect to a given price-predicate.

Definition 4.9. (ϕ-p-safety)
Let $b \in \mathbb{N}^k_\omega$ and ϕ be a price-predicate. We say that the prcwf-net N is *$\phi(b)$-p-safe* for $m_0 \in P_S^{\oplus}$ if for each $j > 0$, every run of N starting in m_0^j satisfies $\phi(b)$.

Definition 4.10. (ϕ-dynamic soundness)
Let $b \in \mathbb{N}^k_\omega$ and ϕ be a price-predicate. We say that the prcwf-net N is $\phi(b)$-*dynamically sound* for $m_0 \in P^{\oplus}_S$ if for each $j > 0$ and for each marking m reachable from m_0^j by firing some r_1, we can reach a marking $m_f \in \mathcal{M}^j_{out}$ by firing some r_2 such that $r_1 \cdot r_2$ satisfies $\phi(b)$.

Ordinary dynamic soundness is obtained by taking ϕ as the constantly true predicate. As we have mentioned before, prices are different from resources in that they do not constraint the behavior of the net. However, once we are interested in checking a priced-soundness problem, it is natural to consider the available “budget” as an extra resource. Indeed, this can be done but only for firing costs, which are local to transitions, but again this is not possible for storage costs.

Let us see some simple facts about ϕ-p-safety and ϕ-dynamic soundness.

Proposition 4.1. The following facts hold:

1. If $\phi_1 \to \phi_2$ holds, then $\phi_1(b)$-p-safety implies $\phi_2(b)$-p-safety, and $\phi_1(b)$-dynamic soundness implies $\phi_2(b)$-dynamic soundness.
2. For any ϕ, ϕ-dynamic soundness implies (unpriced) dynamic soundness.
3. In general, ϕ-dynamic soundness is undecidable for rcwf-nets without the path property.

Proof:
(1) is straightforward by Def. 4.9 and Def. 4.10. (2) follows from (1), considering that any ϕ entails the constantly true predicate. (3) follows from the undecidability of (unpriced) dynamic soundness for rcwf-nets without the path property [24]. $\square$

As a (not very interesting) example, if ϕ is the constantly false price-predicate, no prcwf-net is ϕ-dynamically sound, so that it is trivially decidable. Now we factorize ϕ-dynamic soundness into unpriced dynamic soundness and p-safety. As we proved in the previous section, if we consider negative costs safety is undecidable even for priced wf-nets. Therefore, for now on we will focus in rcwf-nets with non-negative costs.

Proposition 4.2. Let ϕ be a price-predicate and N a prcwf-net with non-negative costs. Then N is $\phi(b)$-dynamically sound if and only if it is dynamically sound and $\phi(b)$-p-safe.

Proof:
First notice that for any run r of N and any run r' extending r we have $\phi(b, \mathcal{P}(r \cdot r')) \to \phi(b, \mathcal{P}(r))$. Indeed, it is enough to consider that, because we are considering that costs are non-negative, $\mathcal{P}(r) \leq^{\oplus} \mathcal{P}(r \cdot r')$ holds and, by Def. 4.8, ϕ is downward closed in its second parameter. For the if-part, if N is dynamically sound and all its runs satisfy $\phi(b)$ then it is clearly $\phi(b)$-dynamically sound. Conversely, if it is $\phi(b)$-dynamically sound it is dynamically sound by Prop. 4.1. Assume by contradiction that there is a run r that does not satisfy $\phi(b)$. By the previous observation, no extension of r can satisfy $\phi(b)$, so that N is not $\phi(b)$-dynamically sound, thus reaching a contradiction. $\square$

Therefore, to decide ϕ-dynamic soundness we can consider those two properties separately. Though (unpriced) dynamic soundness is undecidable for rcwf-nets (without the path property), it is decidable for the subclass of proper rcwf-nets [24]. In the following sections, we will study the decidability of $\phi(b)$-p-safety for various price-predicates, even if N is not proper.

To conclude this section, and as we did in the previous one, notice that for any price-predicate ϕ, the set $\mathcal{B}_\phi(N) = \{b \in \mathbb{N}^k_\omega \mid N$ is $\phi(b)$-dynamically sound ($\phi(b)$-p-safe)$\}$ is upward-closed because of the upward-closure in the first parameter of price-predicates. Therefore, and as we did for pwf-nets, we can apply the Valk & Jantzen result to compute the minimal budgets b for which N is $\phi(b)$-p-safe ($\phi(b)$-dynamically sound) whenever we can decide $\phi(b)$-p-safety ($\phi(b)$-dynamic soundness) for each $b \in \mathbb{N}^k_\omega$.

5. Selected price predicates

Now, we study some specific cases of these price predicates. In particular, we study the maximum, the sum, the average and the discounted sum.

5.1. Sum and Max-dynamic soundness

Let us now study the two first of the concrete priced problems for prcwf-nets. When we consider several instances of a workflow net running concurrently, we may be interested in the overall accumulated price, or in the highest price that the execution of each instance may cost.

If we consider again the prcwf-net in Fig. 4, the price of a student completing the exercise is unbounded, since an unbounded amount of students could be in p_4, and because t_2 can go wrong an unbounded number of times (it is possible to loop in a cycle with positive cost). This corresponds to the idea that the prcwf-net is not $Max(b)$-p-safe for any b. This situation could be fixed by allowing a maximum number of retries per student in step two, and by limiting the maximum number of students that are waiting in p_4, which can be done using static places.

Definition 5.1. (Sum and Max price-predicates)
We define the price-predicates Sum and Max as:

$$\begin{aligned} Sum(b, A) &\iff \textstyle\sum_{x\in A} x \leq b \\ Max(b, A) &\iff x \leq b \text{ for all } x \in A \end{aligned}$$

Sum and Max are indeed price-predicates because they satisfy the conditions in Def. 4.8. They are both upward closed in the first parameter and downward closed in the second. Let us remark that the cost model given by Sum, in which all the prices are accumulated, is the analogous to the cost models in [5, 6]. However, since we are here interested in the behavior of an arbitrary number of instances, a necessary condition for $Sum(b)$-p-safety is that all instances, except for a finite number of them, have a null price (for those components in b that are not ω).

Example 5.1. Consider the prcwf-net N in Fig. 3, and a run of N with n instances, and in which t_2 is not fired until t_1 has been fired n times. The price of the i-th instance in any such run is $2(i-1)$. Indeed, the first firing of t_1 costs nothing, because there are no tokens in p, but in the second one there is already a token in p, so that the second firing costs 2 (because $S(p, t_1) = 2$). In particular, the last instance of the net costs $2(n-1)$. Therefore, the net is neither $Max(b)$-p-safe nor $Sum(b)$-p-safe for any $b \in \mathbb{N}$.

Now, suppose that $S(q,t) = 0$ for each place q and transition t, $C(t_1) = 1$ and $C(t_2) = 0$. Each instance costs exactly 1, so that it is $Max(1)$-p-safe. However, in a run in which n instances have reached out, the sum of the prices of all instances is n, so the net is not $Sum(b)$-p-safe for any $b \in \mathbb{N}$.

Now we prove decidability of Max and Sum-p-safety by reducing them to non-coverability problems in a $w\nu$-PN. Given a prcwf-net N we build a $w\nu$-PN $\mathcal{C}(N)$, the *cost representation net* of N, by adding to N new places, whose tokens represent the costs of each run. Then, the net will be safe iff no marking with $b_i + 1$ tokens in the place representing the i^{th} component of the prices can be covered. We simulate firing costs by adding to N "normal arcs", without whole-place operations, but for the simulation of storage costs we need the whole-place capabilities of $w\nu$-PN.

Proposition 5.1. Max-p-safety and Sum-p-safety are decidable for prcwf-nets, even without the path property. Max-dynamic soundness and Sum-dynamic soundness are decidable for proper prcwf-nets.

Proof:
By Prop. 4.2, it is enough to consider the price-safety problems. We reduce Sum-p-safety to coverability for $w\nu$-PN. Then, we show how to adapt this reduction to the case of Max-p-safety. Let $N = (P, T, F, H, C, S)$ be a prcwf-net with price arity k and $b \in \mathbb{N}^k_\omega$. We can assume that b has no ω-components, or we could safely remove the cost information of those components. We build the $w\nu$-PN $\mathcal{C}(N)$ as follows: First, we consider a transition t_ν that can put at any time a fresh name in the place in, to simulate the fact that in N the initial marking can have an arbitrary amount of different names in that place. We also consider new places $c_1, ..., c_k$ to compute the costs of the current run. If the instance a fires a transition t, then $C(t)[i]$ a-tokens are put in c_i (this can be done with regular postarcs). Moreover, for each place p in N, the current marking of p is copied $S(p,t)[i]$ times in c_i (this is done thanks to the whole-place operations of $\mathcal{C}(N)$). Then, at any point, the number of tokens in $c_1, ..., c_k$ represents the cost of the current run. In order to reduce Sum-p-safety to a coverability problem, it is enough to allow the possibility that every token in c_i carries the same name. This can be done by adding a transition t_{c_i} that takes two different tokens from c_i and puts one of them twice back in c_i. Let us now formalize the previous ideas. We define $\mathcal{C}(N) = (P^c, T^c, F^c, G^c, H^c)$ as:

- $P^c = P \cup \{c_1, ..., c_k\}$,
- $T^c = T \cup \{t_\nu\} \cup \{t_{c_1}, ..., t_{c_k}\}$.
- For each $t \in T$,
  - $F^c_t(p) = F_t(p)$ if $p \in P$, and $F^c_t(p) = \emptyset$, otherwise,
  - $H^c_t(p) = H_t(p)$ if $p \in P$, and $H^c_t(c_i) = C(t)[i] * \{x\}$, otherwise,
  - $G^c_t(p,p') = \begin{cases} S(p,t)[i] & \text{if } p \in P, \text{ and } p' = c_i, \\ 1 & \text{if } p = p', \\ 0 & \text{otherwise.} \end{cases}$
- For each $i \in \{1, ..., k\}$,
  - $F^c_{t_{c_i}}(c_i) = \{x, y\}$, and $F^c_{t_{c_i}}(p) = \emptyset$ otherwise,

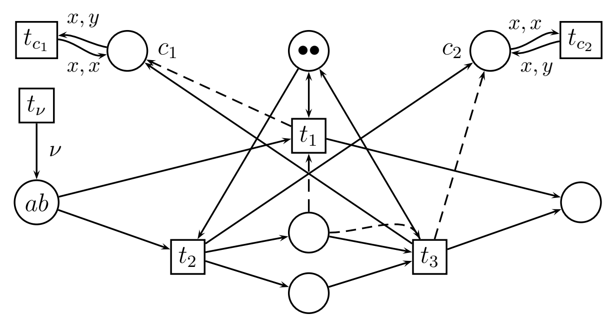

Figure 5. The costs representation $w\nu$-PN of the prcwf-net in Fig. 2

  - $H^c_{t_{c_i}}(c_i) = \{x, x\}$, and $H^c_{t_{c_i}}(p) = \emptyset$ otherwise, and
  - $G^c_{t_{c_i}}$ is the identity matrix.

- $F_{t_\nu}(p) = \emptyset$ for any $p \in P^c$, $H_{t_\nu}(in) = \{\nu\}$ and returns the empty multiset elsewhere, and G_{t_ν} is the identity matrix.

Any run r of N can be simulated by a run of $\mathcal{C}(N)$, preceded by several firings of t_ν. Moreover, if r starts in m_0 and finishes in m (seen as a run of $\mathcal{C}(N)$), then by construction of $\mathcal{C}(N)$ it holds that the sum of the prices of the instances in r, is the vector formed by considering the number of tokens (maybe with different colors) in $c_1, ..., c_k$. In particular, when a transition in T is fired, we add the corresponding firing costs to places c_i by ”normal arcs”, that is, we have $H^c_t(c_i) = C(t)[i] * \{x\}$. Moreover, we add storage costs by copying the necessary number of times the tokens in dynamic places to places c_i, that is, we have $G^c_t(p, c_i) = S(p, t)[i]$. Finally, as each transition t_{c_i} takes two tokens with different names from c_i, and puts them back, changing the name of one of them by the name of the other token, these transitions allow to reach the markings in which the sum of the prices of all the instances of a run is represented by the tokens in the places c_i, all of them with the same name. Then, N is $Sum(b)$-p-unsafe if and only if there is $j \in \{1, ..., k\}$ such that the marking with $b[j] + 1$ tokens of the same color in c_j and empty elsewhere is coverable, and we are done.

The previous construction with some modifications also yields decidability of $Max(b)$-p-safety. We add one more place $last$ (which will always contain the name of the last instance that has fired a transition) and for each $i \in \{1, ..., k\}$, we add a new place d_i (where we will compute the costs). When a transition $t \in T$ is fired, in $\mathcal{C}(N)$ we replace the name in $last$ by the name of the current transition, and reset every place c_i (by setting $G_t(c_i, c_i) = 0$). Moreover, we change the effect of every t_{c_i}: they now take a token from c_i, and put a token of the name in $last$ in the place d_i (see Fig. 6).

Therefore, when a transition $t \in T$ is fired, it is possible to reach a marking in which the costs of firing t are added to every d_i (represented by the name of the instance that has fired t) by firing t followed by the firing of every t_{c_i} n_i times, provided t put n_i tokens in c_i. Notice that if another transition fires before, then that run is lossy, in the sense that it is computing an underapproximation of its cost, but it is always possible to compute the exact cost. Therefore, N is $Max(b)$-p-unsafe iff there is $j \in \{1, ..., k\}$ such that the marking with $b[j] + 1$ tokens of the same color in d_j and empty elsewhere is coverable. $\square$

Example 5.2. Fig. 5 shows the costs representation net of the net N in Fig. 2. For a better readability, we have removed some of the labels of the arcs. As the prices in N are vectors of $\mathbb{N}^2$, we have added two places, c_1 and c_2, to store the costs; and two transitions t_{c_1} and t_{c_2}, which take two tokens of different colors of the corresponding places and put them back, with the same color. Moreover, we have added

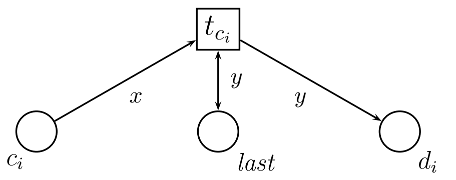

Figure 6. The mechanism added for Max

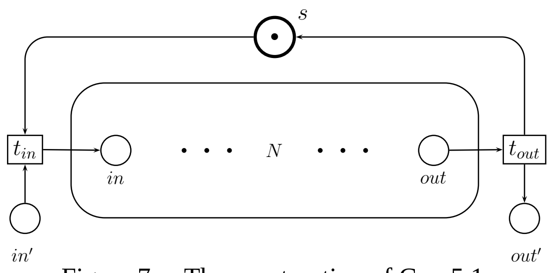

Figure 7. The construction of Cor. 5.1

arcs that manage the addition of the cost of transitions. In particular, dashed arcs denote copy arcs, meaning that when the corresponding transition is fired, tokens are copied in the places indicated by the arrows (which is the effect of G in the proof of the previous result). Then, $Sum(b)$-p-safety is reduced to non-coverability problems: the prcwf-net is $Sum(1,1)$-p-safe iff neither m_1 (the marking with only two tokens carrying the same name in c_1) neither m_2 (the marking with only two tokens carrying the same name in c_2) are coverable.

We remark that if we consider a cost model in which each instance only pays for its own tokens, as discussed after Def. 4.6, the previous proof can be adapted by considering a version of $w\nu$-PN with finer whole-place operations, which are still a subclass of the ones considered in [4], so that the result would still apply. To conclude this section, we show that we can reduce b-p-soundness and b-p-safety problems for pwf-nets with non-negative costs, defined in Sect. 3, to Max-dynamic soundness for prcwf-nets. Therefore, if we consider non-negative costs, we can prove the decidability of b-p-soundness (already proved in Prop. 3.2 in the case with negative costs but considering that the path property holds), and decidability of b-p-safety.

Corollary 5.1. b-p-safety and b-p-soundness are decidable for pwf-nets with non-negative costs, even without the path property.

Proof:
Let N be a pwf-net. To decide b-p-safety it is enough to build a prcwf-net N' by adding to N a single static place s, initially containing one token, two new places in' and out' (the new initial and final places), and two new transitions t_{in} and t_{out}. Transition t_{in} can move a name from in' to in whenever there is a token in s, that is, $F_{t_{in}}(in') = \{x\}$, $F_{t_{in}}(s) = \{\epsilon\}$ and $F_{t_{in}}$ is empty elsewhere, and $H_{t_{in}}(in) = \{x\}$ and empty elsewhere. Analogously, t_{out} can move a name from out to out', putting the black token back in s, that is, $F_{t_{out}}(out) = \{x\}$, and empty elsewhere, and $H_{t_{out}}(out') = \{x\}$, $H_{t_{out}}(s) = \{\epsilon\}$, and empty elsewhere (see Fig. 7). In this way, the concurrent executions of N' are actually sequential. Since there is no other way in which instances can synchronize with each other (because there are no more static places) the potential behavior of all instances coincide, and coincide in turn with the behavior of N. Finally, we take the cost of firing t_{in} and t_{out} as null, as well as the cost of storing tokens in in' and out' for any transition, and the cost of storing tokens in any place for t_{in} and t_{out}. More precisely, $C(t_{in}) = C(t_{out}) = \mathbf{0}$, $S(p, t_{in}) = S(p, t_{out}) = \mathbf{0}$ for any $p \in P$, and $S(in', t) = S(out', t) = \mathbf{0}$ for any $t \in T$. In this way, the cost of each instance is the cost of a run of N. Therefore, N is b-p-safe if and only if N' is $Max(b)$-p-safe. Since weak soundness is decidable for wf-nets [3], we conclude. □

5.2. Av-dynamic soundness

Now we study the next of the concrete priced-soundness problems. Instead of demanding that the execution of each instance does not exceed a given budget (though the price of one instance depends on the others), we will consider an amortized, or average price.

Definition 5.2. (Av price-predicate)
We define the price-predicate Av as $Av(b, A) \Leftrightarrow (\sum_{x\in A} x)/\,|A| \leq b$.

Therefore, N is $Av(b)$-p-safe if in average, the price of each instance does not exceed b, for any number of instances. Alternatively, we could have a slightly more general definition, in which we only considered situations in which the number of instances exceeds a given threshold $l > 0$. More precisely: $Av_l(b, A) \Leftrightarrow |A| \geq l \rightarrow (\sum_{x\in A} x)/\,|A| \leq b$. We will work with Av, though we claim that with fairly minor changes in our techniques we could also address the slightly more general price-predicate Av_l.

Example 5.3. Consider the prcwf-net in Fig. 10. The cost of firing t_1 is twice the number of instances in place in when t_1 is fired. Therefore, the net is $Av(2)$-p-safe, though not $Max(b)$-p-safe for any $b \in \mathbb{N}$.

Now suppose that we force t_2 to be fired in the first place, t_1 in second place, and then t_2 as many times as possible, by adding some static conditions. Moreover, consider that the cost of firing t_1 is three times the number of instances in place out (instead of twice the number of instances in in) when t_1 is fired. Then, if we consider any run r with two or more instances, in which we fire t_1 and t_2 in the beginning, the sum of the prices of the instances of r is 3. Therefore, the if we consider such a run with two instances, the average price is $3/2 > 1$, and then the net is not $Av(1)$-p-safe. However, if we consider that the number of instances is greater than two, the average of the prices always remains under 1, and therefore the net is $Av_l(1)$-p-safe if we consider any threshold $l \geq 3$.

Next we prove decidability of Av-dynamic soundness, though the case of Av-p-safety remains open. We can reduce Av-dynamic soundness of a prcwf-net N to (unpriced) dynamic soundness of an rcwf-net N^b. In order to ensure $Av(b)$-p-safety, the maximum budget we may spend in an execution with n instances is $b * n$. Essentially, the idea of this construction is to add to N new places $s_1 \dots s_k$ in which tokens represent the remaining budget, and remove tokens from them when transitions are fired. Moreover, each transition will have $s_1 \dots s_k$ as preconditions, so that if the net has consumed all the budget, then it halts before reaching the final marking. Therefore, we add $b\,[i]$ tokens to s_i each time an instance starts its execution, for each i. The simulation is "lossy" because of how we manage storage costs, but it preserves dynamic soundness. The proof of the next proposition gives a detailed explanation of this construction.

Proposition 5.2. Av-dynamic soundness is decidable for proper prcwf-nets.

Proof:
Let k be the price arity of N. We reduce Av-dynamic soundness to unpriced dynamic soundness. Given a prcwf-net N and $b \in \mathbb{N}^k_\omega$, let us see that there is an rcwf-net N^b such that N is $Av(b)$-dynamically sound if and only if N^b is dynamically sound. We start the construction of N^b by adding to N new static places $s_1, ..., s_k$ that initially contain one token each. These new places store the budget than can be consumed by instances, plus the initial extra token. For that purpose, every instance adds $b[i]$ tokens to s_i when it starts. When a transition t is fired, we remove from s_i $C(t)[i]$ tokens to cope with firing costs.

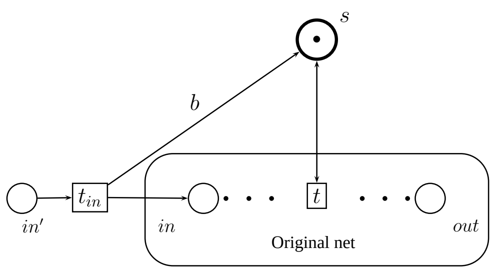

Figure 8. Construction for $Av(b)$-dynamic soundness

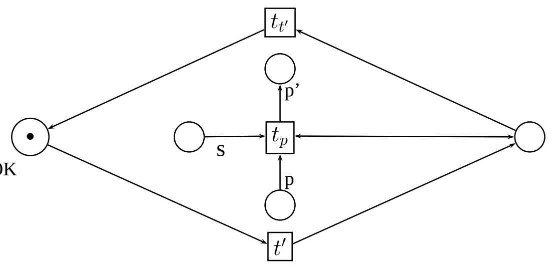

Figure 9. Schema of the managing of storage costs assuming $S(p,t) = 1$

We will later explain how to cope with storage costs (notice that N^b is an rcwf-net, and in particular it does not have whole-place operations). Moreover, each transition has $s_1, ..., s_k$ as preconditions and postconditions. Therefore, the net will deadlock when some s_i is empty, meaning that it has used strictly more than the allowed budget. Then, if N is not $Av(b)$-dynamically sound, N^b halts before reaching the final marking for some execution, and therefore, it is not dynamically sound. Moreover, if N is $Av(b)$-dynamically sound, then N^b is dynamically sound, because each place s_i always contains tokens, and therefore the executions of N^b represent executions of N. Fig. 8 shows a schema of the reduction for price arity 1.

Now we address the simulation of storage costs. Fig. 9 depicts the following construction. We simulate them in a "lossy" way, meaning that if the firing of t in N costs v, in the simulation we will remove *at most* $v[i]$ tokens from s_i. To do that, for each place p of N we will add a new place p'. When a transition t is fired, for each place p we transfer tokens from p to p', one at a time (transition t_p in the figure), removing at each time $S(p,t)[i]$ tokens from s_i. We add the same mechanism for the transfer of tokens from p' to p and some static conditions to make sure that if we have started transferring tokens from p to p' because of the firing of a transition, we do not transfer tokens from p' to p because of the same firing, and the other way around. At any point, the transfer can stop (even if some tokens have not been transfered), which finishes the simulation of t. Since we now have two places representing each place p (p and p'), for each transition of N, we need to add several transitions in order to be able to take (or put) tokens from p, p' or both.

Having lossy computations of the cost of a run, if N exceeds the average budget for some execution and some number of instances, then N^b will have a deadlock when this execution is simulated correctly (meaning that all the tokens which have to be transfered are indeed transfered). Then, N^b is not dynamically sound. Conversely, if N is $Av(b)$-dynamically sound (and in particular no run of N exceeds the average budget), then N^b never consumes all the tokens in any s_i, and it behaves as N, so that it is dynamically sound. □

5.3. Ordered prices

So far, we have considered that instances are not ordered in any way, following directly the approaches in [19, 20, 24]. Nevertheless, we could consider an order between the instances, and use it to compute the price of a run in such a way that the relative order between instances matter. A sensible way to do that is to assume a linear order between instances within a run given by the order in which they start their

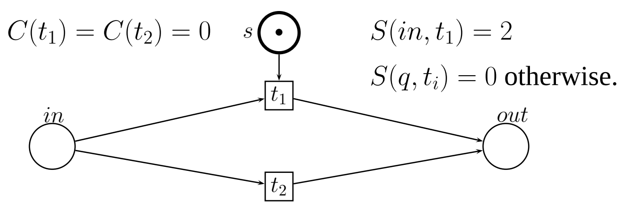

Figure 10. Av-p-safety does not imply Max-p-safety

execution.

Definition 5.3. (Order between instances)
Let N be a prcwf-net, and a and b be two instances in a run r of N. We write $a <_r b$ if a is removed from in in r before b, and $a =_r b$ if neither a nor b have been removed from in in r. We write $a \leq_r b$ if $a =_r b$ or $a <_r b$.

Then, the order $\leq_r$ is a total order over the set of instances in r. In this situation we can write $Id(r) = a_1 \leq_r \cdots \leq_r a_n$ to denote that $a_1, ..., a_n$ are all the instances in r, ordered as indicated. In the following, for a set A we denote by A^* the set of finite words over A.

Definition 5.4. (Ordered price of a run)
Given a run r of a prcwf-net with $Id(r) = a_1 \leq_r \cdots \leq_r a_n$ we define the *ordered price* of r as the word $\mathcal{P}_o(r) = \mathcal{P}(a_1, r)...\mathcal{P}(a_n, r) \in (\mathbb{N}^k)^*$.

Notice that the previous definition is correct in the sense that whenever $a =_r b$ then we have $\mathcal{P}(a, r) = \mathcal{P}(b, r) = 0$. Moreover, the instances of a run are always ordered as $a_1 <_r \cdots <_r a_m < a_{m+1} =_r ... =_r a_n$.

With the notion of ordered price, we can consider price-predicates that depend on the order in which instances are fired. Therefore, ordered price-predicates are predicates over $\mathbb{N}^k_\omega \times (\mathbb{N}^k)^*$. We consider the order $\leq^*$ over $(\mathbb{N}^k)^*$ given by $w_1...w_n \leq^* w_1...w'_m$ iff $n \leq m$ and for each $0 < i \leq n$, $w_i \leq w'_i$. For instance, following [10], we can model situations in which costs in the future are less important than closer ones.

Definition 5.5. (Ds-price predicate)
Given $0 < \lambda < 1$, we define the *discounted-sum* price-predicate Ds_λ as

$$Ds_\lambda(b, v_1...v_n) \Leftrightarrow \sum_{i=1}^{n} \lambda^i * v_i \leq b$$

Example 5.4. Let us recall the run of the net N of Fig. 3 described in Ex. 5.1. We proved that the net is neither $Max(b)$-p-safe nor $Sum(b)$-p-safe for any $b \in \mathbb{N}$. Moreover, the average price of the run is $\sum_{i=1}^{n} 2(i-1)/n$, which equals $n - 1$, so that it is not $Av(b)$-p-safe for any $b \in \mathbb{N}$. However, the discounted price of the run is $\sum_{i=1}^{n} 2(i-1)\lambda^i$, with $0 < \lambda < 1$. By using standard techniques, it can be seen that the limit of those sums is $b = 2\lambda^2/(1-\lambda)^2$. Moreover, for $\lambda = 1/c$ with $c > 1$ that formula simplifies to $2/(c-1)^2$. As it is easy to prove that the considered runs are the most expensive ones of N, it follows that it is $Ds_\lambda(b)$-p-safe for that $b \in \mathbb{N}$.

Note that if we consider $\leq^*$, then Ds_λ is downward-closed in its second argument. Decidability of Ds_λ-p-safety remains open, but a weaker version of this problem, in which we only consider finitely many instances, is decidable.

Definition 5.6. (Fds-price predicate)
Given $0 < \lambda < 1$ and $l \in \mathbb{N}$, we define the finite-discounted-sum price-predicate

$$Fds^l_\lambda(b, v_1...v_n) \Leftrightarrow \sum_{i=1}^{min\{n,l\}} \lambda^i * v_i \leq b$$

For this finite version of discounted-sum, p-safety is decidable.

Proposition 5.3. Let $l \in \mathbb{N}$, $c \in \mathbb{N} \setminus \{0\}$ and $\lambda = 1/c$. Fds^l_λ-p-safety is decidable for prcwf-nets. Fds^l_λ-dynamic soundness is decidable for proper prcwf-nets.

Proof:
We reduce Fds^l_λ-p-safety to coverability for $w\nu$-PN. Basically, we build a new net, in which the first l instances are managed separately, in order to store their weighted prices in places as in the proof of Prop. 5.1. We consider $l+1$ copies of the net, one for each of the first l instances, and one for the rest of the instances, to give the proper weight to the prices that we store. Let $N = (P, T, F, H, C, S)$ be a prcwf net. Let us build a new $w\nu$-PN $N' = (P', T', F', G', H')$ as follows:

For each dynamic place $p \in P$, we consider $p, p_1, \ldots, p_l$ in P', and for each $t \in T$, we take $t, t_1, \ldots, t_l$ in T'. If $p \neq in$ then, for each $i \in \{1, \ldots, l\}$, $F'_t(p) = F'_{t_i}(p_i) = F_t(p)$ and $H'_t(p) = H'_{t_i}(p_i) = H_t(p)$. For each static place $s \in P$, we consider $s \in P'$. Moreover, for each $t \in T$ and for each $i \in \{1, \ldots, l\}$, $F'_t(s) = F'_{t_i}(s) = F_t(s)$ and $H'_t(s) = H'_{t_i}(s) = H_t(s)$. Therefore N' has $l+1$ copies of the dynamic part of N, sharing the static part.

We manage separately the places $in, in_1, \ldots, in_l$, in order to make sure that the i^{th} copy of the dynamic part of N corresponds with the i^{th} instance that has started. In particular, when we remove a token from a place in_i, we put a token in in_{i+1}, to allow the next instance to start. Given $t \in T$ such that $F_t(in) \neq \emptyset$ then, for each $i \in \{1, \ldots, l\}$, $F'_t(in_i) = F'_{t_i}(in) = F_t(in)$ and for each $i \in \{1, \ldots, l-1\}$, $H'_{t_i}(in_{i+1}) = \nu$. Finally, the last instances will be managed in the same last copy of the net, in which places do not have indexes, so $H'_{t_l}(in) = H'_t(in) = \nu$.

Now that we have the structure for the different copies of the net, we add some places to store the weighted prices of the first l instances. Let n be the arity of b. Then, $pr_1, \ldots, pr_n \in P'$ will be places where we store the prices. To give a weight to each instance, we consider the following:

$\sum_{i=1}^l \lambda^i * v_i \leq b \Leftrightarrow \sum_{i=1}^l 1/c^i * v_i \leq b \Leftrightarrow c^l(\sum_{i=1}^l 1/c^i * v_i) \leq c^l * b \Leftrightarrow \sum_{i=1}^l c^{l-i} * v_i \leq c^l * b$.

Then, for each i, we will store the prices of the i^{th} instance with weight c_j^{l-i} as in the proof of Prop. 5.1, that is, we define G' and H' for places pr_i,considering G'_{t_i} and H'_{t_i} as G'_t and H'_t in Prop. 5.1, multiplied by c^{l-i}. Therefore if we prove that for each run of N' and each place pr_j the total stored price in pr_j is less than $c^i * b_j$, the predicate $Fds^l_\lambda(b)$ will hold for the net N.

Finally, as in Prop. 5.1, we add transitions to make all the tokens in each pr_j carry the same name, and we just need to check that for no $j \in \{1, ..., n\}$ the marking m_j with $m_j(pr_j) = \{a^{c^l * b_j + 1}\}$ and $m_j(p) = \emptyset$ otherwise (for an arbitrary $a \in Id$) is coverable in N', to conclude that N is $Fds^l_\lambda(b)$-p-safe.

$\square$

5.4. Complexity

Now, we would like to study the complexity of the previous problems. As a preliminary result, we study the complexity of the safety property for the defined priced predicates, in the case in which the path property does not necessarily hold. More precisely, we reduce coverability for ν-PN to each of the previous safety problems, and therefore, they are at least non primitive-recursive. For that purpose, we introduce the single-name coverability problem, which has the same complexity as coverability, and we set that this problem can be reduced to the problem of deciding ϕ-p-safety, for each of our predicates.

Definition 5.7. We define the single-name-coverability problem as that of given a ν-PN N with initial marking m_0, and m_f a marking with a single identifier, deciding whether m_f can be covered in N.

Single-name-coverability is a problem more restricted than coverability, which is decidable for ν-PN [28]. Next we prove that its complexity is the same as that of general coverability.

Lemma 5.1. The single-name-coverability problem has non primitive recursive complexity.

Proof:
We reduce coverability, which has non primitive recursive complexity [28], to single-name-coverability. Let m_f be the final marking. It is enough to add a new transition that can be fired whenever m_f is covered, thus putting a (black) token in a new place. Thus, m_f can be covered in the original ν-PN iff the marking with a black token in the new place can be covered. Moreover, the reduction is clearly polynomial, so we are done. $\square$

Proposition 5.4. The single-name-coverability problem for ν-PNs can be reduced in polynomial time to each of the following problems for prcwf-nets without the path property: $Max(1)$-p-safety, $Sum(1)$-p-safety, $Av(1)$-p-safety and $Ds_{1/c}(1)$-p-safety (with $c \in \mathbb{N} \setminus \{0\}$).

Proof:
Let N be a ν-PN, and a marking m_f of N with a single name. In order to reduce the coverability of m_f to the previous problems in polynomial time, we first build a new rcwf-net N' which simulates the behavior of N, and then, we will add the prices in four different ways, building four different prcwf-nets, N_1, N_2, N_3 and N_4, in order to reduce coverability of m_f to $Max(1)$-p-safety, $Sum(1)$-p-safety, $Av(1)$-p-safety and $Ds_\lambda(1)$-p-safety, respectively. In fact, the marking m_f of N can be covered if and only if the previous problems have a negative answer.

Let us first build the rcwf-net N', by adding to N a new place in, to be used as a fresh name storage. Then, when a transition creating a fresh name is fired in N, in N' a name is taken from in, and put in the corresponding places. Therefore, any run of N can be simulated by a run of N' with enough fresh tokens initially in the place in. Moreover, we add to N' a new transition t_f which has the marking m_f as precondition, as well as a new place p in which every name that is taken from in is stored. For more details on this part of the construction see the proof of undecidability of dynamic soundness in [24].

Finally, we assign the prices to N'. The only transition with storage or firing costs is t_f. Then, a run r will have a price greater than zero if and only if t_f is fired in r, so that m_f can be covered if and only if there is a run of N', with a price greater than zero. Now we define the three different prices, which reduce coverability to $Max(1)$-p-safety, $Sum(1)$-p-safety, $Av(1)$-p-safety and $Ds(1)$-p-safety, respectively.

- $Max(1)$-p-safety: $C(t_f) = 2$ and $S(q, t_f) = 0$ for each place q. If m_f is covered in N, there is a run r of N' in which t_f is fired, and then, the price of r is 2, which is greater than 1. Conversely, if m_f is not covered in N, t_f cannot be fired in any run of N', and the price of any run is 0.
- $Sum(1)$-p-safety: $C(t_f) = 2$ and $S(q, t_f) = 0$ for each place q. This case is analogous to the previous one.
- $Av(1)$-p-safety: $C(t_f) = 0$, $S(p, t_f) = 2$ and $S(q, t_f) = 0$ for each place $q \neq p$. If m_f is covered in N, there is a run r of N' in which t_f is fired by an instance once. The price of the instance which has fired t_f is $2n$, where n is the number of instances which have started in r. The average sum of the price of r is $2 * n/n = 2 > 1$. The other implication is as in the case of $Max(1)$-p-safety.
- $Ds_{1/c}(1)$-p-safety: $C(t_f) = 0$, $S(p, t_f) = c$ and $S(q, t_f) = 0$ for each place $q \neq p$. The proof is analogous to the previous one.

□

By the previous polynomial reduction, and because single-name-coverability has non primitive recursive complexity, we can conclude the following.

Corollary 5.2. $Max(1)$-p-safety, $Sum(1)$-p-safety, $Av(1)$-p-safety and $Ds(1)$-p-safety have non primitive recursive complexity for prcwf-nets without the path property.

5.5. Relating price predicates

In this subsection we study the relations between the previous price predicates. More precisely, for each pair of predicates ϕ and ψ, we will study whether $\phi(b)$-dynamic soundness entails $\psi(b')$-dynamic soundness for some b'. Moreover, we will try to set the relation between these two bounds.

First of all, let us focus on the prcwf-nets of Fig. 3 and Fig 10. As we showed in the previous examples, the first of these nets proves that a net may be $Ds_\lambda(b)$-dynamically sound for a certain b, but not Av-dynamically sound, Max-dynamically sound nor Sum-dynamically sound for any bound. Analogously, the second net is $Av(b)$-dynamically sound for some $b \in \mathbb{N}$, but not Ds_λ-dynamically sound, Max-dynamically sound nor Sum-dynamically sound for any bound.

The next propositions set the remaining relations:

Proposition 5.5. Let N be a $Sum(b)$-dynamically sound prcwf-net for some $b \in \mathbb{N}$. Then, N is also $Max(b)$-dynamically sound, $Av(b)$-dynamically sound and $Ds_\lambda(b)$-dynamically sound.

Proof:
If we prove $Sum(b) \to Max(b), Av(b), Ds_\lambda(b)$, we are done (Prop. 4.1). Let $A = \{x_1, \ldots, x_n\}$ be a set of prices satisfying $Sum(b)$, that is, $\sum_{x \in A} x \leq b$. Then, since we are considering non-negative prices, for all $a \in A$, $a \leq \sum_{x \in A} x \leq b$, and therefore A satisfies $Max(b)$. Moreover, $\sum_{x \in A} x/n \leq \sum_{x \in A} x \leq b$ and if $0 < \lambda < 1$ then $\sum_{i=1}^{n} \lambda^i * x_i < \sum_{x \in A} x \leq b$. Then A satisfies $Av(b)$ and $Ds_\lambda(b)$ too. □

Proposition 5.6. Let N be a $Max(b)$-dynamically sound prcwf-net for some $b \in \mathbb{N}$. Then, N is also $Av(b)$-dynamically sound and $Ds_\lambda(b')$-dynamically sound, where $b' = \lambda * b/(1 - \lambda)$. In particular, N is $Ds_{1/2}(b)$-dynamically sound.

	Sum	**Max**	**Ds**	**Av**
Sum	✓	✓	✓	✓
Max	× (Ex. 5.1)	✓	☑	✓
Ds	× (Fig. 3)	× (Fig. 3)	✓	× (Fig. 3)
Av	× (Fig. 10)	× (Fig. 10)	× (Fig. 10)	✓

Table 1. A ✓ symbol in row ϕ_1 and column ϕ_2 means that $\phi_1(b)$-dynamic soundness implies $\phi_2(b)$-dynamic soundness; a ☑ symbol means that the implication holds for a possibly different b; a × means that the implication does not hold

Proof:
Let us prove that $Max(b) \rightarrow Av(b)$ and $Max(b) \rightarrow Ds_\lambda(b')$. Let $A = \{x_1, \ldots, x_n\}$ be a set of prices which satisfies $Max(b)$, that is, for all $x \in A$, $x \leq b$. Let $m = max\{x_1, \ldots, x_n\}$. Then, $\sum_{x \in A} x/n \leq n * m/n = m \leq b$ and therefore A satisfies $Av(b)$. Moreover, if $0 < \lambda < 1$ then $\sum_{i=1}^{n} \lambda^i * x_i \leq \sum_{i=1}^{n} \lambda^i * m \leq \sum_{i=1}^{n} \lambda^i * b \leq \lambda * b/(1-\lambda)$, so A satisfies $Ds_\lambda(b')$. □

Table 1 summarizes the relations between the different predicates.

6. Conclusions and open problems

We have extended the study of workflow processes, adding prices to them. In particular, we have added firing and storage costs to wf-nets and rcwf-nets, as done for priced Petri nets in [5]. Then, we have defined priced versions of safety and soundness for pwf-nets, and several notions of the same properties for rcwf-nets, depending on how we aggregate local prices to obtain a global price.

The main decidability results regarding pwf-nets we have proved here are:

- b-p-safety is undecidable when negative costs are considered.
- b-p-safety and b-p-soundness are both decidable for non-negative costs.

As b-p-safety for pwf-nets can be easily reduced to Sum, Max, Av and Fds-p-safety for prcwf-nets, we have not considered negative costs in these cases. For prcwf-nets, our main results are:

- Sum, Max, and Fds-p-safety are decidable.
- Sum, Max, Av and Fds-dynamic soundness are decidable for the subclass of proper prcwf-nets.

There are interesting open problems that remain open, as the decidability of Av-p-safety and the problems related to the discounted sum. Their study would be a good starting point for the study of more sophisticated aggregation techniques, like the Gini or the Theil indices [13].

More study regarding the complexity of the problems studied here is needed. As a preliminary result, we have showed that coverability for ν-PN (which has a non-primitive recursive complexity [28]) can be reduced to Sum, Av and Max safety for nets without the path property, so that they are non-primitive recursive. Further research is needed to investigate the complexity of safety properties for nets with the path property, and soundness properties.

A way in which we must extend this work is to consider that storage costs depend on how long tokens stay on places during the firing of transitions. For this purpose, time for rcwf-nets should be considered instead of the arbitrary interleavings in the firing of concurrent transitions, as done in [6, 14, 15]. Then, priced safety and soundness properties could be studied in this timed model.

Finally, the study of Ds-soundness, leads us to several interesting questions about how the size of the markings and prices of a (sound) rcwf-net may grow. In this sense, we would be interested in studying possible bounds for the number of tokens in places, or for the costs of an instance in terms of the number of instances running in the net.

References

[1] van der Aalst, W. M. P.: Verification of Workflow Nets, *ICATPN* (P. Azéma, G. Balbo, Eds.), 1248, Springer, 1997, ISBN 3-540-63139-9.

[2] van der Aalst, W. M. P., van Hee, K. M.: *Workflow Management: Models, Methods, and Systems*, MIT Press, 2002, ISBN 0-262-01189-1.

[3] 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: classification, decidability, and analysis, *Formal Asp. Comput.*, **23**(3), 2011, 333–363.

[4] Abdulla, P. A., Delzanno, G., Begin, L. V.: A classification of the expressive power of well-structured transition systems, *Inf. Comput.*, **209**(3), 2011, 248–279.

[5] Abdulla, P. A., Mayr, R.: Minimal Cost Reachability/Coverability in Priced Timed Petri Nets, *FOSSACS* (L. de Alfaro, Ed.), 5504, Springer, 2009, ISBN 978-3-642-00595-4.

[6] Abdulla, P. A., Mayr, R.: Computing Optimal Coverability Costs in Priced Timed Petri Nets, *LICS*, IEEE Computer Society, 2011, ISBN 978-0-7695-4412-0.

[7] Alur, R., Torre, S. L., Pappas, G. J.: Optimal paths in weighted timed automata, *Theor. Comput. Sci.*, **318**(3), 2004, 297–322.

[8] Bouyer, P., Brihaye, T., Bruyère, V., Raskin, J.-F.: On the optimal reachability problem of weighted timed automata, *Formal Methods in System Design*, **31**(2), 2007, 135–175.

[9] Bouyer, P., Fahrenberg, U., Larsen, K. G., Markey, N.: Timed automata with observers under energy constraints, *HSCC* (K. H. Johansson, W. Yi, Eds.), ACM ACM, 2010, ISBN 978-1-60558-955-8.

[10] Chatterjee, K., Doyen, L., Henzinger, T. A.: Quantitative languages, *ACM Trans. Comput. Log.*, **11**(4), 2010.

[11] Chatterjee, K., Doyen, L., Henzinger, T. A., Raskin, J.-F.: Generalized Mean-payoff and Energy Games, *FSTTCS* (K. Lodaya, M. Mahajan, Eds.), 8, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010, ISBN 978-3-939897-23-1.

[12] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C.: *Introduction to Algorithms*, 2 edition, The MIT Press, 2001.

[13] Cowell, F.: *Measuring Inequality*, Oxford University Press, 2011.

[14] Desel, J., Erwin, T.: Modeling, Simulation and Analysis of Business Processes, *Business Process Management* (W. M. P. van der Aalst, J. Desel, A. Oberweis, Eds.), 1806, Springer, 2000, ISBN 3-540-67454-3.

[15] Desel, J., Erwin, T.: Quantitative Engineering of Business Processes with VIP business, *Petri Net Technology for Communication-Based Systems* (H. Ehrig, W. Reisig, G. Rozenberg, H. Weber, Eds.), 2472, Springer, 2003, ISBN 3-540-20538-1.

[16] Dickson, L. E.: Finiteness of the Odd Perfect and Primitive Abundant Numbers with n Distinct Prime Factors, *American Journal of Mathematics*, **35**(4), 1913, pp. 413–422, ISSN 00029327.

[17] Fahrenberg, U., Juhl, L., Larsen, K. G., Srba, J.: Energy Games in Multiweighted Automata, *ICTAC* (A. Cerone, P. Pihlajasaari, Eds.), 6916, Springer, 2011, ISBN 978-3-642-23282-4.

[18] Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere!, *Theor. Comput. Sci.*, **256**(1-2), 2001, 63–92.

[19] van Hee, K. M., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Soundness of Resource-Constrained Workflow Nets, *ICATPN* (G. Ciardo, P. Darondeau, Eds.), 3536, Springer, 2005, ISBN 3-540-26301-2.

[20] Juhás, G., Kazlov, I., Juhásová, A.: Instance Deadlock: A Mystery behind Frozen Programs, *Petri Nets* (J. Lilius, W. Penczek, Eds.), 6128, Springer, 2010, ISBN 978-3-642-13674-0.

[21] Lazic, R., Newcomb, T., Ouaknine, J., Roscoe, A. W., Worrell, J.: Nets with Tokens which Carry Data, *Fundam. Inform.*, **88**(3), 2008, 251–274.

[22] Liu, K., Jin, H., Chen, J., Liu, X., Yuan, D., Yang, Y.: A Compromised-Time-Cost Scheduling Algorithm in SwinDeW-C for Instance-Intensive Cost-Constrained Workflows on a Cloud Computing Platform, *IJHPCA*, **24**(4), 2010, 445–456.

[23] Magnani, M., Montesi, D.: BPMN: How Much Does It Cost? An Incremental Approach, *BPM* (G. Alonso, P. Dadam, M. Rosemann, Eds.), 4714, Springer, 2007, ISBN 978-3-540-75182-3.

[24] Martos-Salgado, M., Rosa-Velardo, F.: Dynamic Soundness in Resource-Constrained Workflow Nets, *FMOODS/FORTE* (R. Bruni, J. Dingel, Eds.), 6722, Springer, 2011, ISBN 978-3-642-21460-8.

[25] Martos-Salgado, M., Rosa-Velardo, F.: Cost Soundness for Priced Resource-Constrained Workflow Nets, *Petri Nets* (S. Haddad, L. Pomello, Eds.), 7347, Springer, 2012, ISBN 978-3-642-31130-7.

[26] Mukherjee, D.: *QoS in WS-BPEL Processes*, Ph.D. Thesis, Indian Institute Of Technology, 2008.

[27] Rosa-Velardo, F., de Frutos-Escrig, D.: Name Creation vs. Replication in Petri Net Systems, *Fundam. Inform.*, **88**(3), 2008, 329–356.

[28] Rosa-Velardo, F., de Frutos-Escrig, D.: Decidability and complexity of Petri nets with unordered data, *Theor. Comput. Sci.*, **412**(34), 2011, 4439–4451.

[29] Sampath, P., Wirsing, M.: Computing the Cost of Business Processes, *UNISCON* (J. Yang, A. Ginige, H. C. Mayr, R.-D. Kutsche, Eds.), 20, Springer, 2009, ISBN 978-3-642-01111-5.

[30] Tahamtan, A., Oesterle, C., Tjoa, A. M., Hameurlain, A.: BPEL-TIME - WS-BPEL Time Management Extension, *ICEIS (3)* (R. Zhang, J. Cordeiro, X. Li, Z. Zhang, J. Zhang, Eds.), SciTePress, 2011, ISBN 978-989-8425-55-3.

[31] Valk, R., Jantzen, M.: The Residue of Vector Sets with Applications to Decidability Problems in Petri Nets, *Acta Inf.*, **21**, 1985, 643–674.