Cost Soundness for Priced 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. We extend workflow Petri nets (wf-nets) with discrete prices, by associating a price to the execution of a transition and, more importantly, to the storage of tokens. We first define the soundness problem for priced wf-nets, 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 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 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 each all instances are considered. We study the relations between these properties, together with their decidability.

1 Introduction

Workflow nets (wf-nets) are an important formalism for the modeling of business processes, or workflow management systems [1, 2]. Roughly, a wf-net is a Petri net with two special places, *in* and *out*. Its initial 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 no supervisor is needed in order to ensure the completion of the task under a fairness assumption. 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 [2].

Recent works [3–5] 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 singe instance of an rcwf-net is sound, several instances could deadlock because of these shared resources. In [3] the authors define dynamic soundness, the condition

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

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 number of resources is as in the initial state. The paper [4] 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 [5] we continued the work in [4], 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 dynamic soundness 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 [6–10]. 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.

In order to study these problems, in this paper we add prices to our nets, similarly as done for the (untimed) priced Petri nets in [11]. We consider different types of prices, modeled as tuples of integers or naturals. More precisely, we add firing costs to transitions, and more importantly, 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 while 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 essentially we can always reach the final marking, without spending more than a given budget. For priced rcwf-nets, the definition of soundness is not so straightforward, since it must consider the behavior of an arbitrary number of instances. We consider a parametric definition of soundness for priced rcwf-nets. For any run, we collect the prices of every instance in the run, so that soundness is 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, the sum, the average and the discounted sum.

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 [12, 13]. In these cases we have decidability of priced soundness within the proper subclass. As a

corollary, 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, price-safety remains open.

Related Work. 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. In [14] 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 [11, 15, 16]. E.g., the paper [11] 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 [17]. 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.

Quantitative aspects of reactive systems are studied as energy games e.g. in [18–20]. For example, in [20] 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. Sect. 2 gives some notations we will use throughout the paper. Sec. 3 extends wf-nets with prices and proves undecidability of price-safety with negative costs. In Sect. 4 we extend rcwf-nets and give some basic results. In Sect. 5 we study some specific cases of price predicates. Finally, in Sect. 6 we present our conclusions. Missing proofs can be found in [21].

2 Preliminaries

A quasi-order $\leq$ over a set A is a reflexive and transitive binary relation over A. Given a quasi-order $\leq$, we say 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 and $\mathbf{0} = (0, ..., 0)$. We write $v[i]$ to denote the i^{th} component of $v \in \mathbb{N}_\omega^k$. 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. 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 ${}^\bullet in = \emptyset$, $in^\bullet \neq \emptyset$, ${}^\bullet out \neq \emptyset$ and $out^\bullet = \emptyset$,
- for each $p \in P \setminus \{in, out\}$, ${}^\bullet p \neq \emptyset$ and $p^\bullet \neq \emptyset$.

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 [2]. 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 allow whole-place operations and broadcasts, similar to Data Nets [13]. 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 [13] in which names can be created fresh. When a transition t of a $w\nu$-PN is fired, four operations

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

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 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 $p \in P$, $m'(p) = \sum_{p' \in P}((m(p') - \sigma(F_t(p'))) * G_t(p', p)) + \sigma(H_t(p))$, and we denote this by $m \overset{t(\sigma)}{\to} m'$.

Example 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 labelled 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 [12] 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 [22].

Proposition 1. *Coverability is decidable for wν-PN.*

Finally, we define ν-PN [23], 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 be most of the time 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 very 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 [11]. 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 during the firing of each transition, respectively.

Definition 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 can 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 define the price of a transition.

Definition 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 $r = m_1 \xrightarrow{t_1} m_2 \xrightarrow{t_2} m_3 \dots m_n \xrightarrow{t_n} m_{n+1}$ *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 [11]. However, storage costs cannot be simulated by firing costs, since the former are marking dependent, while the latter are not. Next, we define safeness of a pwf-net with respect prices.

Definition 3 (b-p-safeness). *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-safe if all the runs that reach the final marking cost less than the given budget. Next, define soundness for pwf-nets.

Definition 4 (b-soundness). *Given $b \in \mathbb{N}_\omega^k$, we say that a pwf-net is* b-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 $b \in \mathbb{N}_\omega^k$. It is easy to see that a pwf-net is b-sound iff it is weakly sound and b-p-safe. However, as we prove next, the latter is undecidable.

Proposition 2. *b-p-safeness is undecidable.*

Proof (sketch). We reduce the cost-threshold-reachability problem for PPN with negative costs, which is undecidable [11]. A PPN is a P/T net endowed with storage and firing costs. The cost-threshold-reachability problem consists in, given m_f and $b \in \mathbb{N}_\omega^k$, decide whether there is a run σ with $m_0 \xrightarrow{\sigma} m_f$ such that $C(\sigma) \leq v$. The reduction consists essentially in obtaining the pwf-net as the *inverse* of the PPN, by taking the inverse of each firing and storage cost. Then, if a run σ has price c in the PPN, the simulating run has cost $v - c$, so that it satisfies the cost-threshold-reachability property iff $N \not\models \mathbf{0}$. The details of the proof can be found in [21].

Instead of addressing the problem of b-soundness with non-negative costs directly, 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 the results regarding pwf-nets as a corollary of the more general problem.

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

Theorem 1 ([24]). *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-sound), provided we can decide b-p-safety (b-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 [5]. The definition we use is equivalent to those in [3,

4], though more convenient for our purposes. 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 [25]. 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 $x, \epsilon \in Var \backslash \Upsilon$.

Definition 5 (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 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 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 6 (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 labelled by ϵ, and arcs to/from dynamic places are labelled by x). N_p, the production net of N, is the P/T net obtained by projecting N to its dynamic places.

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. Now we define the priced version of rcwf-nets, analogously as in Def. 1.

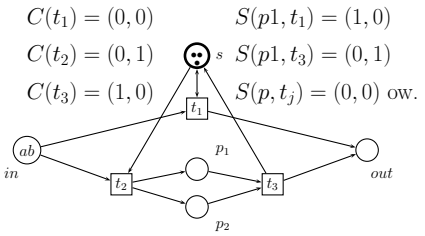

Fig. 2. A priced resource-constrained workflow net

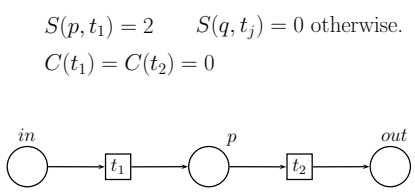

Fig. 3. prcwf-net not $Sum(b)$ neither $Max(b)$-dynamically sound for any $b \in \mathbb{N}$

Definition 7 (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 associated. We start by defining the price of an instance in a run.

Definition 8 (Price of an instance). *We define the* price of an instance $a \in Id(m_0)$ *in a run* $r = m_0 \xrightarrow{t_1(\sigma_1)} m_1 \xrightarrow{t_2(\sigma_2)} m_2 \dots m_{n-1} \xrightarrow{t_n(\sigma_n)} m_n$ *of a prcwf-net as*

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

Intuitively, we are considering those transitions in r fired by a, and computing its price as we did in Def. 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 penalitation 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.

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.

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

Definition 9 (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-safeness and dynamic soundness. More precisely, those properties for prcwf-nets are parameterized with respect to a price-predicate.

Definition 10 (Price-predicate). *A* price-predicate ϕ *of arity* $k \geq 0$ *is a predicate over* $\mathbb{N}_\omega^k \times (\mathbb{Z}^k)^\oplus$ *such that if* $b \leq b'$ *and* $A' \leq^\oplus A$ *then* $\phi(b, A) \rightarrow \phi(b', A')$ *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 $b \in \mathbb{N}_\omega^k$, 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-safeness and dynamic soundness, though with respect to a given price-predicate.

Definition 11 (ϕ-p-safeness). *Let* $b \in \mathbb{N}_\omega^k$ *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 12 (ϕ-dynamic soundness). *Let* $b \in \mathbb{N}_\omega^k$ *and* ϕ *be a price-predicate. We say that the prcwf-net* N *is* $\phi(b)$-dynamically sound *for* $m_0 \in P_S^\oplus$ *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* $m_f \in \mathcal{M}_{out}^j$ *by firing some* r_2 *such that* $r_1 \cdot r_2$ *satisfies* $\phi(b)$*.*

Ordinary dynamic soundness [5] is obtained by taking ϕ as the constantly true predicate. Let us see some simple facts about ϕ-p-safeness and ϕ-dynamic soundness.

Proposition 3. *The following holds:*

1. *If* $\phi_1 \rightarrow \phi_2$ *holds, then* $\phi_1(b)$*-p-safeness implies* $\phi_2(b)$*-p-safeness, 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.*

Proof. (1) is straightforward by Def. 11 and Def. 12. (2) follows from (1), considering that any ϕ entails the constantly true predicate. (3) follows from the undecidability of (unpriced) dynamic soundness for general rcwf-nets [5]. □

Therefore, ϕ-dynamic soundness is undecidable for some ϕ, though certainly not for all. 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 safeness is undecidible even for priced wf-nets. Therefore, for now on we will focus in rcwf-nets with costs in $\mathbb{N}$.

Proposition 4. *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 $\phi(b, \mathcal{P}(r \cdot r')) \rightarrow \phi(b, \mathcal{P}(r))$. 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. 10, 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. 3. 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. □

Therefore, to decide ϕ-dynamic soundness we can consider those two properties separately. Though (unpriced) dynamic soundness is undecidable for general rcwf-nets, it is decidable for a subclass of rcwf-nets that we call proper rcwf-nets [5]. An rcwf-net is proper if its production net is weakly sound and for each transition t in the production net, $t^\bullet \neq \emptyset$. The first condition can be intuitively understood as the rcwf-net behaving properly (being weakly sound) if endowed with infinitely-many resources, which amounts to removing the restriction of its behavior by means of resources. The second condition is a slight relaxation of the standard connection condition considered for wf-nets [2]: for any $n \in P \cup T$ there exists a path from *in* to n and from n to *out*. That is because, given a transition $t \in T$, if there exists a path from t to *out*, then there must be $p \in P$ with $p \in ({}^\bullet t)$ and therefore $t^\bullet \neq \emptyset$.

In turn, it is decidable to check that an rcwf-net is proper [5]. In the following sections, we will study the decidability of $\phi(b)$-p-safeness 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 $\mathcal{B}_\phi(N) = \{b \in \mathbb{N}^k_\omega \mid N \text{ is } \phi(b)\text{-dynamically sound } (\phi(b)\text{-p-safe})\}$ 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-safeness ($\phi(b)$-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.

Definition 13 (*Sum* and *Max* price-predicates). *We define the price-predicates Sum and Max as:*

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

Sum and *Max* are indeed price-predicates because they satisfy the conditions in Def. 10. 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 [11, 17]. However, since we are here interested in the behavior of an arbitrary number of instances, a necessary condition for $\mathit{Sum}(b)$-p-safeness is the existence of an instance from which the rest of the instances have a null price (for those components in b that are not ω).

Example 2. 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 $2 \cdot (i-1)$. 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 $2 \cdot (n-1)$. Therefore, the net is neither $\mathit{Max}(b)$-p-safe nor $\mathit{Sum}(b)$-p-safe for any $b \in \mathbb{N}$.

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

Now we prove decidability of $\mathit{Max}(b)$ and $\mathit{Sum}(b)$-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. More precisely, we will use a-tokens to compute the cost of the instance represented by a. 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. *Max-p-safety and Sum-p-safety are decidable for prcwf-nets. Max-dynamic soundness and Sum-dynamic soundness are decidable for proper prcwf-nets.*

Proof. We reduce *Sum*-p-safety to coverability for $w\nu$-PN. Then, we sketch 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. Let $b \in \mathbb{N}^k_\omega$. 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 $\mathcal{C}(N) = (P^c, T^c, F^c, G^c, H^c)$ as follows:

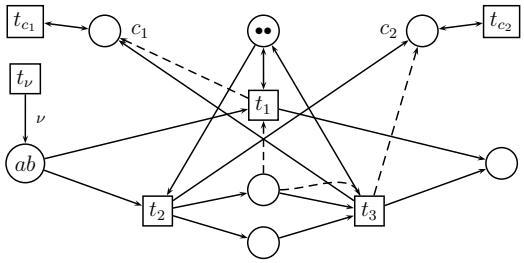

Fig. 4. The costs representation $w\nu$-PN of the prcwf-net in Fig. 2

- $P^c = P \cup \{c_1, ..., c_k\}$,
- $T^c = T \cup \{t_\nu\} \cup \{t_{c_1}, ..., t_{c_k}\}$.
- 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,
  - $H^c_{t_{c_i}}(c_i) = \{x, x\}$, 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 $c_1, ..., c_k$. 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 each markings in which the sum of the prices of all 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 not coverable, and we are done.

The previous construction with some modifications also yields decidability of $Max(b)$-p-safeness. 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. 5).

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 if and only if 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 not coverable. □

Example 3. Fig. 4 shows the costs representation net of the net N in Fig. 2. For a better readability, we have removed 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 colours of the correspondig places and put them back, with the same colour. Moreover, we have added arcs that manage the addition of the costs 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-safeness 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. 8, 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 [12], so that the result would still apply. To conclude this section, we show that we can reduce the soundness problem for pwf-nets defined in Sect. 3 to *Max*-dynamic soundness for prcwf-nets.

Corollary 1. *b-p-safeness and b-soundness are decidable for pwf-nets with non-negative costs.*

Proof. Let N be a pwf-net. To decide b-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. 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,

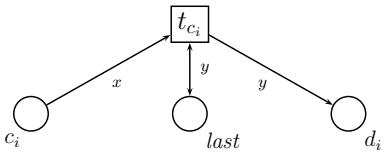

Fig. 5. The mechanism added for *Max*

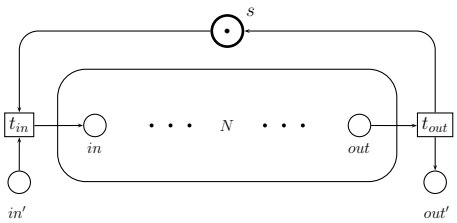

Fig. 6. The construction of Cor. 1

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 [2], 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 14 (*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 4. Consider the prcwf-net in Fig. 9. 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 it is not $Max(b)$-p-safe for any $b \in \mathbb{N}$.

Now suppose that we force t_1 to be fired in the first place by adding some static conditions. Then, though the net is not $Av(0)$-p-safe, it is $Av_l(0)$-p-safe if we consider any threshold $l \geq 2$.

We can reduce $Av(b)$-dynamic soundness of a prcwf-net N to (unpriced) dynamic soundness of an rcwf-net N^b. In order to ensure $Av(b)$-p-safeness, 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 in which tokens

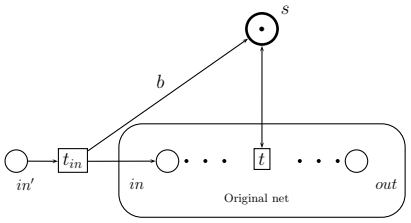

Fig. 7. Construction for $Av(b)$-dynamic soundness

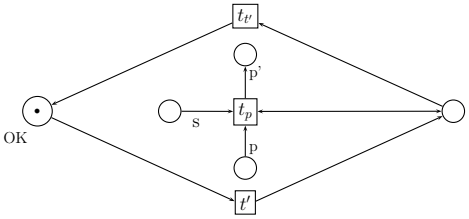

Fig. 8. Schema of the managing of storage costs assuming $S(p,t) = 1$

represent the remaining budget, and remove tokens from them when transitions are fired. Moreover, each transition will have s as a precondition, so that if the net has consumed all the budget, then it halts before reaching the final marking. Therefore, we add b tokens to s each time an instance starts its execution. 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 6. *Given a prcwf-net N and $b \in \mathbb{N}_\omega^k$, there is an rcwf-net N^b such that N is $Av(b)$-dynamically sound if and only if N^b is dynamically sound.*

Proof. Let k be the price arity of N. 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. In order to obtain that, 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. 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. 7 shows a schema of the reduction for price arity 1.

Now we address the simulation of storage costs. Fig. 8 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. At any point, the transfer can stop (even if some tokens have not been

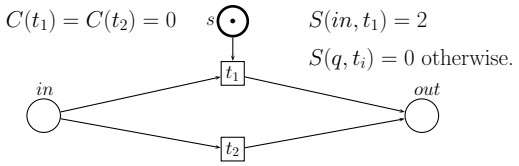

Fig. 9. *Av*-p-safety does not imply *Max*-p-safety

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. □

Corollary 2. *Av-dynamic soundness is decidable for proper prcwf-nets.*

5.3 Ordered prices

So far, we have considered that instances are not ordered in any way, following directly the approaches in [3–5]. 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 execution.

Definition 15 (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 16 (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 $a_1 <_r \cdots <_r a_m < a_{m+1} =_r ... =_r a_n$.

	Sum	**Max**	**Ds**	**Av**
Sum	✓	✓	✓	✓
Max	× (Ex. 2)	✓	⍁ [21]	✓ [21]
Ds	× (Fig. 3)	× (Fig. 3)	✓	× (Fig. 3)
Av	× (Fig. 9)	× (Fig. 9)	× (Fig. 9)	✓

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 possibly for a different b; a × means that the implication does not hold

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 $n < m$ and for each $0 < i \leq n$, $w_i \leq w'_i$. For instance, following [14], we can model situations in which costs in the future are less important than closer ones.

Definition 17 (*Ds*-price predicate). *Given* $0 < \lambda < 1$*, we define the* discounted-sum *price-predicate* Ds_λ *as* $Ds_\lambda(b, v_1...v_n) \Leftrightarrow \sum_{i=1}^{n} \lambda^i \cdot v_i \leq b$.

Example 5. Let us recall the run of the net N of Fig. 3 described in Ex. 2. 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-safeness remains open, but a weaker version of this problem, in which we only consider finitely many instances, is decidable.

Definition 18 (*Fds*-price predicate). *Given* $0 < \lambda < 1$ *and* $k \in \mathbb{N}$*, we define the finite-discounted-sum price-predicate* $Fds^k_\lambda(b, v_1...v_n) \Leftrightarrow \sum_{i=1}^{min\{n,k\}} \lambda^i \cdot v_i \leq b$

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

Proposition 7. *Let* $k \in \mathbb{N}$*,* $c \in \mathbb{N} \setminus \{0\}$ *and* $\lambda = 1/c$*.* Fds^k_λ*-p-safety is decidable for prcwf-nets.* Fds^k_λ*-dynamic soundness is decidable for proper prcwf-nets.*

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 [11]. 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. Table 1 shows the relations between the different predicates. Some of its results are either trivial or proved in [21].

We have proved that b-p-safety is undecidable when negative costs are considered, but decidable for non-negative costs. Moreover, b-soundness is also decidable in this case. Regarding prcwf-nets, we have proved that *Sum*, *Max*, and *Fds*-p-safety are decidable, and *Sum*, *Max*, *Av* and *Fds*-dynamic soundness are decidable for the subclass of proper prcwf-nets.

There are interesting open problems that remain open: decidability of *Av*-p-safety and that of 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 index.

There are several ways in which we can extend this work. It would be interesting to consider that storage costs depend on how long tokens stay on places during the transitions. For this purpose, time for rcwf-nets should be considered instead of arbitrary interleavings in the firing of transitions, as done in [17]. The, priced safety and soundness properties could be studied in this timed model.

Moreover, it would be interesting to study the complexity of the problems studied here. It is easy to see that coverability for ν-PN (which has a non-primitive recursive complexity [23]) can be reduced to *Sum* and *Max* safety, so that they are non-primitive recursive. Further research is needed to investigage the complexity for the remaining predicates.

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., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT Press (2002)
2. 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** (2011) 333–363
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. Martos-Salgado, M., Rosa-Velardo, F.: Dynamic soundness in resource-constrained workflow nets. In Bruni, R., Dingel, J., eds.: FMOODS/FORTE. Volume 6722 of Lecture Notes in Computer Science., Springer (2011) 259–273

6. Sampath, P., Wirsing, M.: Computing the cost of business processes. In Yang, J., Ginige, A., Mayr, H.C., Kutsche, R.D., eds.: UNISCON. Volume 20 of Lecture Notes in Business Information Processing., Springer (2009) 178–183
7. Magnani, M., Montesi, D.: Bpmn: How much does it cost? an incremental approach. In Alonso, G., Dadam, P., Rosemann, M., eds.: BPM. Volume 4714 of Lecture Notes in Computer Science., Springer (2007) 80–87
8. Tahamtan, A., Oesterle, C., Tjoa, A.M., Hameurlain, A.: Bpel-time - ws-bpel time management extension. In Zhang, R., Cordeiro, J., Li, X., Zhang, Z., Zhang, J., eds.: ICEIS (3), SciTePress (2011) 34–45
9. 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** (2010) 445–456
10. Mukherjee, D.: QoS in WS-BPEL Processes. PhD thesis, Indian Institute Of Technology (2008)
11. 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
12. Abdulla, P.A., Delzanno, G., Begin, L.V.: A classification of the expressive power of well-structured transition systems. Inf. Comput. **209** (2011) 248–279
13. Lazic, R., Newcomb, T., Ouaknine, J., Roscoe, A.W., Worrell, J.: Nets with tokens which carry data. Fundam. Inform. **88** (2008) 251–274
14. Chatterjee, K., Doyen, L., Henzinger, T.A.: Quantitative languages. ACM Trans. Comput. Log. **11** (2010)
15. Alur, R., Torre, S.L., Pappas, G.J.: Optimal paths in weighted timed automata. Theor. Comput. Sci. **318** (2004) 297–322
16. Bouyer, P., Brihaye, T., Bruyère, V., Raskin, J.F.: On the optimal reachability problem of weighted timed automata. Formal Methods in System Design **31** (2007) 135–175
17. Abdulla, P.A., Mayr, R.: Computing optimal coverability costs in priced timed petri nets. In: LICS, IEEE Computer Society (2011) 399–408
18. Bouyer, P., Fahrenberg, U., Larsen, K.G., Markey, N.: Timed automata with observers under energy constraints. In Johansson, K.H., Yi, W., eds.: HSCC, ACM ACM (2010) 61–70
19. Chatterjee, K., Doyen, L., Henzinger, T.A., Raskin, J.F.: Generalized mean-payoff and energy games. In Lodaya, K., Mahajan, M., eds.: FSTTCS. Volume 8 of LIPIcs., Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2010) 505–516
20. Fahrenberg, U., Juhl, L., Larsen, K.G., Srba, J.: Energy games in multiweighted automata. In Cerone, A., Pihlajasaari, P., eds.: ICTAC. Volume 6916 of Lecture Notes in Computer Science., Springer (2011) 95–115
21. Martos-Salgado, M., Rosa-Velardo, F.: Cost soundness for priced resource-constrained workflow nets (2012) `http://antares.sip.ucm.es/frosa/`.
22. Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere! Theor. Comput. Sci. **256** (2001) 63–92
23. Rosa-Velardo, F., de Frutos-Escrig, D.: Decidability and complexity of petri nets with unordered data. Theor. Comput. Sci. **412** (2011) 4439–4451
24. Valk, R., Jantzen, M.: The residue of vector sets with applications to decidability problems in petri nets. Acta Inf. **21** (1985) 643–674
25. Rosa-Velardo, F., de Frutos-Escrig, D.: Name creation vs. replication in petri net systems. Fundam. Inform. **88** (2008) 329–356

Appendix: missing proofs

Proposition 2. *b-p-safeness is undecidable.*

Proof. We reduce the cost-threshold-reachability problem for PPN with negative costs, and price arity 1, which is undecidable [11]. 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 [11] that this problem is undecidable.

Given a PPN $N = (P, T, F, C)$ and $b \in \mathbb{N}$, let us build a pwf-net N' as follows. First, we add two new places, *in* and *out*, a transition t_0 that removes a token from *in* and sets the initial marking of N, and a transition t_f that has m_f as precondition and puts a token in *out*. The new places have no storage cost, t_0 has firing cost $b + 1$ and t_f has 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 σ is a run in N with cost c, then $t_0 \cdot \sigma$ is a run in N' with cost $b + 1 - c$. Let us see that there exists a run r of N such that $m_0 \xrightarrow{r} m \geq m_f$ with $C(r) \leq b$ iff N' is not 0-p-safe. Let r be a run such that $m_0 \xrightarrow{r} m \geq m_f$ and $C(r) \leq b$. Let us call $r' = t_0 r t_f$ 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 $t_0 r t_f$, $C(r') = 1 + b - C(r) \leq 0$. Therefore, N' is b-p-sound. □

Proposition 8. *Let $b \in \mathbb{N}_\omega^k$ and N a $Max(b)$-p-safe prcwf-net. Then, N is also $Av(b)$-p-safe and $Ds_\lambda(\lambda * b/(1-\lambda))$-p-safe. In particular, N is $Ds_{1/2}(b)$-p-safe.*

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