Verifying generalised and structural soundness of workflow nets via relaxations*

Michael Blondin[1] [0000−0003−2914−2734], Filip Mazowiecki[2] [0000−0002−4535−6508], and Philip Offtermatt[1,2] [0000−0001−8477−2849]

[1] Université de Sherbrooke, Sherbrooke, Canada
[2] Max Planck Institute for Software Systems, Saarbrücken, Germany

Abstract. Workflow nets are a well-established mathematical formalism for the analysis of business processes arising from either modeling tools or process mining. The central decision problems for workflow nets are k-soundness, generalised soundness and structural soundness. Most existing tools focus on k-soundness. In this work, we propose novel scalable semi-procedures for generalised and structural soundness. This is achieved via integral and continuous Petri net reachability relaxations. We show that our approach is competitive against state-of-the-art tools.

1 Introduction

Workflow nets are a well-established mathematical formalism for the description of business processes arising from software modelers and process mining (*e.g.*, see [2,3]), and further notations such as UML activity diagrams [4]. More precisely, a workflow net consists of *places* that contain resources, and *transitions* that can consume, create and move resources concurrently. Two designated places, denoted i and f, respectively model the initialization and completion of a process. Workflow nets, which form a subclass of Petri nets, enable the automatic formal verification of business processes. For example, 1-*soundness* states that from the initial configuration $\{\mathsf{i}\colon 1\}$, every reachable configuration can reach the final configuration $\{\mathsf{f}\colon 1\}$. Informally, this means that given any partial execution of a business process, it is possible to complete it properly.

Soundness. The main decision problems concerning workflow nets revolve around soundness properties. The generalisation of 1-soundness to several resources is k-*soundness*. It asks whether from $\{\mathsf{i}\colon k\}$, every reachable configuration can reach $\{\mathsf{f}\colon k\}$ (here, $\{p\colon k\}$ indicates that place p contains k resources). Intuitively, 1-soundness guarantees that every initialised process terminates, and k-soundness guarantees that k initialised processes working in parallel will all terminate (see *e.g.* [1,2]). *Generalised soundness* asks whether k-soundness holds for all $k \geq 1$. Unlike k-soundness, generalised soundness preserves desirable properties like

* An extended version of this paper with an appendix containing the missing proofs can be obtained from https://arxiv.org/abs/2206.02606.

composition and has other desirable properties for business applications [21]. *Structural soundness* is the existential counterpart of generalised soundness, *i.e.* it asks whether k-soundness holds for some $k \geq 1$. Structural soundness gives information on how many processes can be controlled in parallel [33], moreover, by applying results about structural soundness, one can compute the set of all k for which the workflow net is k-sound [9, Section 7].

These problems are all decidable [1,22,33], but with high complexity: either PSPACE- or EXPSPACE-complete [9]. Most of the (software) tools focus on k-soundness, with an emphasis on $k = 1$. Existing algorithms for generalised and structural soundness rely on Petri net reachability [22,33,20], which was recently shown Ackermann-complete [25,13], so not primitive recursive. In this work, we describe *novel scalable semi-procedures for generalised and structural soundness.*

We focus on "negative instances", *i.e.* where soundness does *not* hold. Let us motivate this. It is known that given a workflow net $\mathcal{N}$, one can iteratively apply simple reduction rules to $\mathcal{N}$. The resulting workflow net $\mathcal{N}'$ is sound iff $\mathcal{N}$ is as well [10,23]. In practice, one infers that $\mathcal{N}$ is sound from the fact that $\mathcal{N}'$ has been reduced to a trivial workflow net where only i and f remain. However, if $\mathcal{N}$ is *not* sound, one obtains some nontrivial $\mathcal{N}'$ that must be verified via some other approach such as model checking. In this work, we provide algorithmic building blocks for this case, where state-space exploration is prohibitive.

Relaxations. This is achieved by considering two reachability relaxations, namely integer reachability and continuous reachability. As their name suggests, these two notions relax some forbidden behaviour of workflow nets. Informally, integer reachability allows for the amount of resources to become temporarily negative, while continuous reachability allows the fragmentation of resources into pieces. Such relaxations possibly introduce spurious behaviour, but enjoy significantly better algorithmic properties (*e.g.*, see [7]). For example, they have been successfully employed for the verification of multi-threaded program skeletons [16,5,8].

Generalised soundness. Based on these relaxations, we provide two necessary conditions for generalised soundness: *integer boundedness* and *continuous soundness*. The former states that the state-space of a given workflow net is bounded (from above) even under integer reachability. The latter states that a given workflow net is 1-sound under continuous reachability. We show the following for integer boundedness and continuous soundness:

- Well-established classical reduction rules preserve both properties;
- Integer boundedness is testable in polynomial time, and continuous soundness is coNP-complete;
- From a practical viewpoint, they are respectively translatable into instances of linear programming and linear arithmetic (which can be solved efficiently by dedicated tools such as SMT solvers);
- Under a mild computational assumption, continuous soundness implies integer boundedness.

Thus, altogether, in order to check whether a workflow net $\mathcal{N}$ is generalised *unsound*, one may first use classical reduction rules to obtain a smaller workflow net $\mathcal{N}'$; test integer *unboundedness* in polynomial time; and, if needed, move onto testing continuous *unsoundness*.

The fact that continuous reachability can be used to semi-decide generalised soundness is arguably surprising. Using the notation of computation temporal logic (CTL), k-soundness can be rephrased as $\{\mathsf{i}\colon k\} \models \forall\mathsf{G}\,\exists\mathsf{F}\,\{\mathsf{f}\colon k\}$. Some other well-studied properties have a similar structure, *e.g.* liveness and home-stateness amount to "$\boldsymbol{m}_{\text{init}} \models \bigwedge_{t\in T} \forall\mathsf{G}\,\exists\mathsf{F}\,(t \text{ is enabled})$" and "$\boldsymbol{m}_{\text{init}} \models \forall\mathsf{G}\,\exists\mathsf{F}\,\boldsymbol{m}_{\text{home}}$". It is known that liveness, home-stateness, and other properties such as boundedness and inclusion, *cannot* be approximated continuously [8, Sect. 4]. Yet, generalised soundness quantifies k-soundness universally, and this enables a continuous over-approximation. Consequently, we provide a novel application of continuous relaxations for the efficient verification of properties beyond reachability.

Structural soundness. The authors of [33] have observed that a property called structural quasi-soundness is a necessary condition for structural soundness. The former states that $\{\mathsf{i}\colon k\}$ can reach $\{\mathsf{f}\colon k\}$ for some $k \geq 1$. In [33], structural quasi-soundness is reduced to Petri net reachability, which has non primitive recursive complexity. In this work, we show that structural quasi-soundness can be rephrased as continuous reachability. Since the latter can be tested in polynomial time [19], or alternatively via SMT solving [8], this vastly improves the practicability of structural quasi-soundness. We further show that this approach can be adapted so that it provides a lower bound on the first k such that $\{\mathsf{i}\colon k\}$ can reach $\{\mathsf{i}\colon f\}$. From a practical point of view, this is useful as it can vastly reduce the number of reachability queries to decide structural soundness.

Free-choice nets. Many real-world workflow nets have a specific structure where concurrency is restricted. Such nets are known as *free-choice* workflow nets (*e.g.*, see [14] for a book). In particular, free-choice workflow nets allow for the modeling of many features present in common workflow management systems [2]. Generalised soundness is equivalent to 1-soundness for free-choice workflow nets [29]. In this work, we prove that continuous soundness is equivalent to generalised soundness. As a byproduct of our proof, we show that structural soundness is also equivalent to continuous soundness. Altogether, the notions of $\{$1-, generalised, structural, continuous$\}$ soundness *all coincide* for free-choice nets. In particular, this means that the continuous relaxation is *exact* and can serve as an efficient addition to the existing algorithmic toolkit.

Experimental results. To demonstrate the viability of our approach, we have implemented and experimentally evaluated a prototype. As part of our evaluation, we propose several new synthetic instances for generalised and structural soundness, which are hard to decide with naive approaches. Some of these instances involve the composition of workflow nets arising from the modeling of business processes in the IBM WebSphere Business Modeler. Our prototype is competitive against both a state-of-the-art Petri net model checker, and a workflow net analyzer. In particular, our approach exhibits better signs of scalability.

Organization. The paper follows the structure of this introduction. Section 2 introduces notation, workflow nets and some properties. Section 3 defines integer and continuous relaxations, and further shows that they are preserved under reduction rules. Sections 4 to 6 present the aforementioned results on generalised soundness, structural soundness and free-choice nets. Section 7 provides experimental results. Section 8 concludes. Some proofs are deferred to an appendix.

2 Preliminaries

We use $\mathbb{Z}$, $\mathbb{N}$, $\mathbb{Q}$ and $\mathbb{Q}_{\geq 0}$ to respectively denote the integers, the naturals (including 0), the rationals and the nonnegative rationals (including 0). Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Q}^S$ be vectors over a finite set S. We write $\boldsymbol{x} \leq \boldsymbol{y}$ if $\boldsymbol{x}[s] \leq \boldsymbol{y}[s]$ for all $s \in S$. We write $\boldsymbol{x} < \boldsymbol{y}$ if $\boldsymbol{x} \leq \boldsymbol{y}$ and $\boldsymbol{x}[s] < \boldsymbol{y}[s]$ for some $s \in S$. We extend addition and subtraction to vectors, *i.e.* $(\boldsymbol{x}+\boldsymbol{y})[s] \coloneqq \boldsymbol{x}[s]+\boldsymbol{y}[s]$ and $(\boldsymbol{x}-\boldsymbol{y})[s] \coloneqq \boldsymbol{x}[s]-\boldsymbol{y}[s]$ for all $s \in S$. We define $\text{supp}(\boldsymbol{x}) = \{s \in S \mid \boldsymbol{x}[s] \neq 0\}$. Given $c \in \mathbb{Q}$, $\boldsymbol{c} \in \mathbb{Q}^S$ denotes the vector such that $\boldsymbol{c}[s] = c$ for all $s \in S$.

2.1 Petri nets

A *Petri net* $\mathcal{N}$ is a triple (P, T, F), where P is a finite set of *places*; T is a finite set of *transitions*, such that $T \cap P = \emptyset$; and $F \colon ((P \times T) \cup (T \times P)) \to \{0, 1\}$ is a set of *arcs*. For readers familiar with Petri nets, note that arc weights are not allowed, *i.e.* the weights are always 1. A *marking* is a vector $\boldsymbol{m} \in \mathbb{N}^P$ such that $\boldsymbol{m}[p]$ denotes the number of *tokens* in place p. We denote markings listing nonzero values, *e.g.* $\boldsymbol{m} = \{p_1 \colon 1\}$ means $\boldsymbol{m}[p_1] = 1$ and $\boldsymbol{m}[p] = 0$ for $p \neq p_1$.

Let $t \in T$. We define the *pre-vector* of t as ${}^\bullet t \in \mathbb{N}^P$, where ${}^\bullet t[p] \coloneqq F(p, t)$. We define its *post-vector* symmetrically with $t^\bullet[p] \coloneqq F(t, p)$. The *effect* of t is denoted as $\Delta(t) \coloneqq t^\bullet - {}^\bullet t$. We say that a transition t is *enabled* at a marking $\boldsymbol{m}$ if $\boldsymbol{m} \geq {}^\bullet t$. If this is the case, then t can be *fired* at $\boldsymbol{m}$, which results in a marking $\boldsymbol{m}'$ such that $\boldsymbol{m}' \coloneqq \boldsymbol{m} + \Delta(t)$. We write $\boldsymbol{m} \to^t$ to denote that t is *enabled* at $\boldsymbol{m}$, and we write $\boldsymbol{m} \to^t \boldsymbol{m}'$ whenever we care about the marking $\boldsymbol{m}'$ resulting from the firing. We further write $\boldsymbol{m} \to \boldsymbol{m}'$ to denote that $\boldsymbol{m} \to^t \boldsymbol{m}'$ for some $t \in T$.

We say that a sequence of transitions $\pi = t_1 \cdots t_n$ is a *run*. We extend the notion of effect, enabledness and firing from transitions to runs in a straightforward way. The *effect* of a run is defined as the sum of the effects of its transitions, that is, $\Delta(\pi) \coloneqq \Delta(t_1) + \ldots + \Delta(t_n)$. The run π is enabled at $\boldsymbol{m}$, denoted as $\boldsymbol{m} \to^\pi$, if $\boldsymbol{m} \to^{t_1} \boldsymbol{m}_1 \to^{t_2} \boldsymbol{m}_2 \cdots \to^{t_{n-1}} \boldsymbol{m}_{n-1} \to^{t_n}$ for some markings $\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_{n-1}$. Furthermore, firing π from $\boldsymbol{m}$ leads to $\boldsymbol{m}'$, denoted as $\boldsymbol{m} \to^\pi \boldsymbol{m}'$, if $\boldsymbol{m} \to^\pi$ and $\boldsymbol{m}' = \boldsymbol{m} + \Delta(\pi)$. We denote the reflexive and transitive closure of $\to$ by $\to^*$.

A pair $(\mathcal{N}, \boldsymbol{m})$, where $\mathcal{N}$ is a Petri net and $\boldsymbol{m}$ is a marking of $\mathcal{N}$, is called a *marked Petri net*. We write $\text{Reach}(\mathcal{N}, \boldsymbol{m}) \coloneqq \{\boldsymbol{m}' \mid \boldsymbol{m} \to^* \boldsymbol{m}'\}$ to denote the set of markings reachable from $\boldsymbol{m}$ in $\mathcal{N}$.

A marked Petri net $(\mathcal{N}, \boldsymbol{m})$ is *bounded* if there exists $b \in \mathbb{N}$ such that $\boldsymbol{m'} \in \text{Reach}(\mathcal{N}, \boldsymbol{m})$ implies $\boldsymbol{m'}[p] \leq b$ for all $p \in P$. It is further *safe* if $b = 1$. We say *unbounded* and *unsafe* for "not bounded" and "not safe".

Sometimes, we argue about transformations on Petri nets which take as an input a Petri net $\mathcal{N}$ and output a Petri net $\mathcal{N}'$. We say that such a transformation *preserves* some property if $\mathcal{N}$ satisfies that property iff $\mathcal{N}'$ satisfies it.

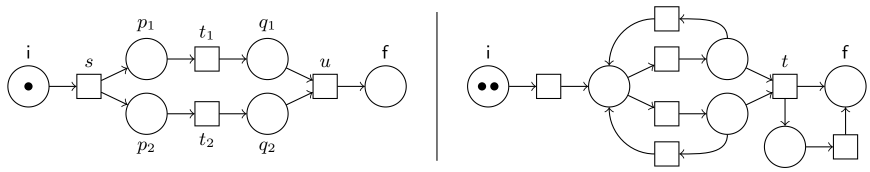

Fig. 1. Example of two Petri nets: respectively $\mathcal{N}_{\text{left}}$ and $\mathcal{N}_{\text{right}}$.

Example 1. The left-hand side of Figure 1 illustrates a Petri net $\mathcal{N}_{\text{left}} = (P, T, F)$ where $P \coloneqq \{\mathsf{i}, p_1, p_2, q_1, q_2, \mathsf{f}\}$, $T \coloneqq \{s, t_1, t_2, u\}$, and F is depicted by arcs, *e.g.* $F[\mathsf{i}, s] = 1$ and $F[s, \mathsf{i}] = 0$. The Petri net is marked by $\{\mathsf{i}\colon 1\}$, *i.e.* with one token in place i. We have $\{\mathsf{i}\colon 1\} \rightarrow^{s} \{p_1\colon 1, p_2\colon 1\} \rightarrow^{t_1 t_2} \{q_1\colon 1, q_2\colon 1\} \rightarrow^{u} \{\mathsf{f}\colon 1\}$. $\lhd$

2.2 Workflow nets

A workflow net $\mathcal{N}$ is a Petri net [1] such that:

- there is a designated *initial place* i such that $t^\bullet[\mathsf{i}] = 0$ for all $t \in T$;
- there is a designated *final place* $\mathsf{f} \neq \mathsf{i}$ such that ${}^\bullet t[\mathsf{f}] = 0$ for all $t \in T$; and
- each place and transition lies on at least one path from i to f in the underlying graph of $\mathcal{N}$, *i.e.* (V, E) where $V \coloneqq P \cup T$ and $(u, v) \in E$ iff $F(u, v) \neq 0$.

We say that $\mathcal{N}$ is:

- *k-sound* if for all $\boldsymbol{m} \in \text{Reach}(\mathcal{N}, \{\mathsf{i}\colon k\})$ it is the case that $\boldsymbol{m} \rightarrow^* \{\mathsf{f}\colon k\}$ [1];
- *generalised sound* if $\mathcal{N}$ is k-sound for all $k \in \mathbb{N}_{\geq 1}$ [21, Def. 3],
- *structurally sound* if $\mathcal{N}$ is k-sound for some $k \in \mathbb{N}_{\geq 1}$ [6].

Example 2. Figure 1 depicts two workflow nets: $\mathcal{N}_{\text{left}}$ and $\mathcal{N}_{\text{right}}$. The former is generalised sound, but the latter is not. Indeed, from $\{\mathsf{i}\colon 1\}$, transition t cannot be enabled (as transitions preserve the sum of all tokens). Both workflow nets are structurally sound. Indeed, $\mathcal{N}_{\text{right}}$ is 2-sound as it is always possible to redistribute the two tokens so that t can be fired in order to reach $\{\mathsf{f}\colon 2\}$. $\lhd$

3 Reachability relaxations

Fix a Petri net $\mathcal{N} = (P, T, F)$. We describe the two aforementioned relaxations.

Integer reachability. An *integral marking* is a vector $\boldsymbol{m} \in \mathbb{Z}^P$. Any transition $t \in T$ is *enabled* in $\boldsymbol{m} \in \mathbb{Z}^P$, and *firing* t leads to $\boldsymbol{m'} \coloneqq \boldsymbol{m} + \Delta(t)$, denoted $\boldsymbol{m} \to_{\mathbb{Z}}^{t} \boldsymbol{m'}$. We define $\boldsymbol{m} \to_{\mathbb{Z}} \boldsymbol{m'}$ and $\boldsymbol{m} \to_{\mathbb{Z}}^{*} \boldsymbol{m'}$ analogously to the standard setting but w.r.t. $\to_{\mathbb{Z}}^{t}$ rather than $\to^{t}$. Similarly, $\mathbb{Z}\text{-Reach}(\mathcal{N}, \boldsymbol{m}) \coloneqq \{\boldsymbol{m'} \in \mathbb{Z}^P \mid \boldsymbol{m} \to_{\mathbb{Z}}^{*} \boldsymbol{m'}\}$. As transitions are always enabled, the order of a firing sequence is irrelevant. In particular, $\boldsymbol{m} \to_{\mathbb{Z}}^{*} \boldsymbol{m'}$ iff there exists $\boldsymbol{x} \in \mathbb{N}^T$ such that $\boldsymbol{m'} = \boldsymbol{m} + \sum_{t \in T} \boldsymbol{x}[t] \cdot \Delta(t)$. Thus, integer reachability amounts to integer linear programming. Moreover, it is NP-complete [12].

Continuous reachability. A *continuous marking* is a vector $\boldsymbol{m} \in \mathbb{Q}_{\geq 0}^{P}$. Let $\lambda \in (0, 1]$. We say that λt is *enabled* in $\boldsymbol{m}$, denoted $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^{\lambda t}$, if $\boldsymbol{m} \geq \lambda \cdot {}^{\bullet}t$. In this context, λ is called the *scaling factor*. Furthermore, we denote by $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^{\lambda t} \boldsymbol{m'}$ that λt is enabled in $\boldsymbol{m}$, and that its *firing* results in $\boldsymbol{m'} \coloneqq \boldsymbol{m} + \lambda \cdot \Delta(t)$. A sequence of pairs of scaling factors and transitions is called a *continuous run*.

The notations $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}} \boldsymbol{m'}$ and $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^{*} \boldsymbol{m'}$ are defined analogously to the discrete case but with respect to $\to_{\mathbb{Q}_{\geq 0}}^{\lambda t}$ rather than $\to^{t}$ (the internal factors λ can differ). Similarly, $\mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \boldsymbol{m}) \coloneqq \{\boldsymbol{m'} \mid \boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^{*} \boldsymbol{m'}\}$ denotes the markings continuously reachable from $\boldsymbol{m}$. For example, for $\mathcal{N}_{\text{left}}$ from Figure 1 and $\pi \coloneqq \frac{1}{2} s \, \frac{1}{4} t_1$, we have $\{\mathsf{i}\colon 1\} \to_{\mathbb{Q}_{\geq 0}}^{\pi} \{\mathsf{i}\colon 1/2, p_1\colon 1/4, p_2\colon 1/2, q_1\colon 1/4\}$. It is known that continuous reachability, namely determining whether $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^{*} \boldsymbol{m'}$, given $\boldsymbol{m}, \boldsymbol{m'} \in \mathbb{Q}_{\geq 0}^{P}$, can be checked in polynomial time [19].

Let us establish the following helpful lemma similar to [19, Lemma 12(1)].

Lemma 1. *Let $\boldsymbol{m}$, $\boldsymbol{m'}$ be continuous markings. It is the case that $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^{*} \boldsymbol{m'}$ iff there exists $b \in \mathbb{N}_{\geq 1}$ such that $b \cdot \boldsymbol{m} \to^{*} b \cdot \boldsymbol{m'}$.*

3.1 Preservation under reduction rules

In [10], the authors present six reduction rules, denoted $R_1, \ldots, R_6$, that generalize the existing reduction rules of [28]. In the following, we show that these reduction rules preserve natural properties for the two reachability relaxations. This means we will be able to check these properties on a reduced workflow net and get the same results as on the original one.

Formally, the rules simplify a given workflow net $\mathcal{N} = (P, T, F)$. In particular, the places of the resulting workflow net $\mathcal{N}' = (P', T, F')$ form a subset of P. Let us fix a domain $\mathbb{D} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}_{\geq 0}\}$ and let $P' \subseteq P$. For ease of notation, we we write $P'' = P \setminus P'$ to denote the (possibly empty) set of removed places. Rules never remove the initial and output places, *i.e.* $\mathsf{i}, \mathsf{f} \in P'$. We denote by $\pi\colon \mathbb{D}^P \to \mathbb{D}^{P'}$ the obvious projection function, and by $\pi_0\colon \mathbb{D}^{P'} \to \mathbb{D}^P$ the "reverse projection" which fills new places with 0. Formally, $\pi_0(\boldsymbol{m})[p'] \coloneqq \boldsymbol{m}[p']$ for all $p' \in P'$ and $\pi_0(\boldsymbol{m})[p''] \coloneqq 0$ for all $p'' \in P''$.

In [10], the authors prove that the rules preserve generalised soundness. This of course implies that they preserve k-soundness for all k. The technical proposition below will be helpful in the forthcoming sections to show the preservation of useful properties based on reachability relaxations.

Proposition 1. *Let $\mathcal{N} = (P, T, F)$ be a workflow net, and let $\mathbb{D} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}_{\geq 0}\}$. Let $\mathcal{N}' = (P', T', F')$ be a workflow net obtained by applying a reduction rule R_i to $\mathcal{N}$, where $P = P' \cup P''$. The following holds.*

- Rule R_1. *We have $P'' = \{p\}$. There exists a nonempty set $R' \subseteq P'$ such that if $\{\mathsf{i}\colon 1\} \to_{\mathbb{D}}^* \boldsymbol{m}$ in $\mathcal{N}$, then $\boldsymbol{m}[p] = \sum_{r \in R'} \boldsymbol{m}[r']$. Moreover, $\boldsymbol{m} \to_{\mathbb{D}}^* \boldsymbol{n}$ in $\mathcal{N}$ iff $\pi(\boldsymbol{m}) \to_{\mathbb{D}}^* \pi(\boldsymbol{n})$ in $\mathcal{N}'$.*
- Rules R_2 and R_3. *We have $P'' = \emptyset$ and $\boldsymbol{m} \to_{\mathbb{D}}^* \boldsymbol{n}$ in $\mathcal{N}$ iff $\boldsymbol{m} \to_{\mathbb{D}}^* \boldsymbol{n}$ in $\mathcal{N}'$.*
- Rules R_4 and R_5. *We have $P'' = \{p\}$. For all $\boldsymbol{m}'$ and $\boldsymbol{n}'$, $\boldsymbol{m}' \to_{\mathbb{D}}^* \boldsymbol{n}'$ in $\mathcal{N}'$ iff $\pi_0(\boldsymbol{m}') \to_{\mathbb{D}}^* \pi_0(\boldsymbol{n}')$ in $\mathcal{N}$. Further, for all $t \in T$ and $p' \in P'$: either ${}^\bullet t[p] = 1$ implies ${}^\bullet t[p'] = 0$; or $t^\bullet[p] = 1$ implies $t^\bullet[p'] = 0$. Also, for $\mathbb{D} \neq \mathbb{Z}$, if $\exists \boldsymbol{m} : \{\mathsf{i}\colon 1\} \to_{\mathbb{D}}^* \boldsymbol{m} \not\to_{\mathbb{D}}^* \{\mathsf{f}\colon 1\}$ holds in $\mathcal{N}$, then $\exists \boldsymbol{m}' : \{\mathsf{i}\colon 1\} \to_{\mathbb{D}}^* \boldsymbol{m}' \not\to_{\mathbb{D}}^* \{\mathsf{f}\colon 1\}$ holds in $\mathcal{N}'$.*
- Rule R_6. *We have $P'' = \{p_2, \ldots, p_k\}$. There exists $p_1 \in P'$ such that for all $\boldsymbol{n} \in P^{\mathbb{D}}$, if $\sum_{i=1}^k \boldsymbol{m}[p_i] = \sum_{i=1}^k \boldsymbol{n}[p_i]$ and $\boldsymbol{n}[p'] = \boldsymbol{m}[p']$ for $p' \in P' \setminus \{p_1\}$, then $\boldsymbol{m} \to_{\mathbb{D}}^* \boldsymbol{n}$. Moreover, if $\boldsymbol{m}[p_i] = \boldsymbol{n}[p_i] = 0$ for $i > 1$, then $\boldsymbol{m} \to_{\mathbb{D}}^* \boldsymbol{n}$ in $\mathcal{N}$ iff $\pi(\boldsymbol{m}) \to_{\mathbb{D}}^* \pi(\boldsymbol{n})$ in $\mathcal{N}'$.*

4 Using relaxations for generalised soundness

In this section, we explain how reachability relaxations can be leveraged in order to semi-decide generalised soundness of workflow nets. More precisely, we state two necessary conditions for a workflow net to be generalised sound: one phrased in terms of integer reachability, and one in terms of continuous reachability. Furthermore, for each condition we: (1) show that it is preserved under reduction rules, and (2) establish its computational complexity. Overall, this means that to conclude that a given workflow net $\mathcal{N}$ is *not* generalised sound, one may first reduce $\mathcal{N}$, and *then* efficiently test for one of these two necessary conditions.

For integer boundedness, we need the mild assumption of nonredundancy. Let $\mathcal{N} = (P, T, F)$ be a workflow net. We say that a place $p \in P$ is *nonredundant*[3] if there exist $k \in \mathbb{N}_{\geq 1}$ and $\boldsymbol{m} \in \mathbb{N}^P$ such that $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}$ and $\boldsymbol{m}[p] \geq 1$. It is known (and simple to see) that redundant places can be removed from a workflow net without changing whether it is generalised sound. Moreover, testing whether a place is nonredundant can be done in polynomial time. Indeed, by Lemma 1, it amounts to testing for the existence of some $\boldsymbol{m} \in \mathbb{Q}_{\geq 0}^P$ such that $\{\mathsf{i}\colon 1\} \to_{\mathbb{Q}_{\geq 0}}^* \boldsymbol{m}$ and $\boldsymbol{m}[p] > 0$. The latter is known as a *coverability* query and it can be checked in polynomial time [19]. Thus, in order to test whether a given workflow net is generalised sound, one can first remove its redundant places. We call a workflow net without redundant places a *nonredundant workflow net*.

4.1 Integer unboundedness

Recall that a marked Petri net $(\mathcal{N}, \boldsymbol{m})$ is *bounded* if there exists $b \in \mathbb{N}$ such that $\boldsymbol{m}' \in \mathrm{Reach}(\mathcal{N}, \boldsymbol{m})$ implies $\boldsymbol{m}' \leq \boldsymbol{b}$. It is well-known that any 1-sound workflow

[3] This notion is adapted from batch workflow nets considered in [22].

net must be bounded from $\{\mathsf{i}\colon 1\}$ [1]. In particular, this means that boundedness is a necessary condition for generalised soundness. However, testing boundedness has extensive computational cost as it is EXPSPACE-complete [11,30]. Consider the relaxed property of *integer boundedness*. It is defined as boundedness, but where "$\boldsymbol{m'} \in \text{Reach}(\mathcal{N}, \boldsymbol{m})$" is replaced with "$\boldsymbol{m'} \in \mathbb{Z}\text{-Reach}(\mathcal{N}, \boldsymbol{m}) \cap \mathbb{N}^P$".

Proposition 2 ([9, Lemma 5.9]). *Let $\mathcal{N}$ be a nonredundant workflow net. If $(\mathcal{N}, \{\mathsf{i}\colon 1\})$ is integer unbounded, then $\mathcal{N}$ is not generalised sound.*

Proposition 3. *The reduction rules from [10] preserve integer unboundedness.*

Next, we establish the complexity of integer unboundedness in two steps. The first step, in the next proposition, shows that testing integer boundedness amounts to a simple condition, independent of the initial marking. The second step shows the condition can be translated into a linear program over $\mathbb{Q}$, rather than $\mathbb{N}$. As a corollary, integer unboundedness is testable in polynomial time.

Proposition 4. *A marked Petri net $(\mathcal{N}, \boldsymbol{m})$ is integer unbounded iff there exists a marking $\boldsymbol{m'} > \boldsymbol{0}$ such that $\boldsymbol{0} \rightarrow^*_{\mathbb{Z}} \boldsymbol{m'}$ (independent of $\boldsymbol{m}$).*

Proof. Let $\mathcal{N} = (P, F, T)$ be a Petri net and let $\boldsymbol{m} \in \mathbb{N}^P$.

$\Rightarrow$) By assumption, there exist $\boldsymbol{m}_0, \boldsymbol{m}_1, \ldots \in \mathbb{Z}\text{-Reach}(\mathcal{N}, \boldsymbol{m}) \cap \mathbb{N}^P$ such that, for every $i \in \mathbb{N}$, it is the case that $m_i \not\leq \boldsymbol{i}$. Since $(\mathbb{N}^P, \leq)$ is well-quasi-ordered, there exist indices $i_0, i_1, \ldots$ such that $\boldsymbol{m}_{i_j} \leq \boldsymbol{m}_{i_k}$ for all $j < k$. Without loss of generality, we can assume that $\boldsymbol{m}_{i_j} < \boldsymbol{m}_{i_k}$ for all $j < k$, as we could otherwise extract such a subsequence. Recall that each $\boldsymbol{m}_{i_\ell} \in \mathbb{Z}\text{-Reach}(\mathcal{N}, \boldsymbol{m})$. Let $\pi_\ell \in T^*$ be such that $\boldsymbol{m} \rightarrow^{\pi_\ell}_{\mathbb{Z}} \boldsymbol{m}_{i_\ell}$. Let $\boldsymbol{x}_\ell \in \mathbb{N}^T$ be the vector such that $\boldsymbol{x}_\ell(t)$ indicates the number of occurrences of transition t in π_ℓ. Since $(\mathbb{N}^T, \leq)$ is well-quasi-ordered, there exist $j < k$ such that $\boldsymbol{x}_j \leq \boldsymbol{x}_k$. Let $\boldsymbol{m'} \coloneqq \boldsymbol{m}_{i_k} - \boldsymbol{m}_{i_j}$ and $\pi \coloneqq \prod_{t \in T} t^{(\boldsymbol{x}_k[t] - \boldsymbol{x}_\ell[t])}$. We have $\boldsymbol{0} \rightarrow^{\pi}_{\mathbb{Z}} \boldsymbol{m'} > \boldsymbol{0}$ as desired since:

$$\begin{aligned} \boldsymbol{m'} = \boldsymbol{m}_{i_k} - \boldsymbol{m}_{i_j} &= (\boldsymbol{m} + \Delta(\pi_k)) - (\boldsymbol{m} + \Delta(\pi_\ell)) = \Delta(\pi_k) - \Delta(\pi_\ell) \\ &= \sum_{t \in T} \boldsymbol{x}_k[t] \cdot \Delta(t) - \sum_{t \in T} \boldsymbol{x}_\ell[t] \cdot \Delta(t) = \sum_{t \in T} (\boldsymbol{x}_k - \boldsymbol{x}_\ell)[t] \cdot \Delta(t) = \Delta(\pi). \end{aligned}$$

$\Leftarrow$) By assumption $\boldsymbol{0} \rightarrow^{\pi}_{\mathbb{Z}} \boldsymbol{m'} > \boldsymbol{0}$. In particular, this means that $\boldsymbol{m} \rightarrow^{\pi}_{\mathbb{Z}} \boldsymbol{m} + \boldsymbol{m'} \rightarrow^{\pi}_{\mathbb{Z}} \boldsymbol{m} + 2\boldsymbol{m'} \rightarrow_{\mathbb{Z}} \cdots$. Therefore, $(\mathcal{N}, \boldsymbol{m})$ is not integer bounded. $\square$

Proposition 5. *A marked Petri net $(\mathcal{N}, \boldsymbol{m})$, where $\mathcal{N} = (P, T, F)$, is integer unbounded iff this system has a solution: $\exists \boldsymbol{x} \in \mathbb{Q}^T_{\geq 0} : \sum_{t \in T} \boldsymbol{x}[t] \cdot \Delta(t) > \boldsymbol{0}$. In particular, given a workflow net $\mathcal{N}$, testing integer boundedness of $(\mathcal{N}, \{\mathsf{i}\colon 1\})$ can be done in polynomial time.*

4.2 Continuous soundness

Let us now introduce a continuous variant of 1-soundness based on continuous reachability. We prove that this variant, which we call *continuous soundness*, is a necessary condition for generalised soundness, and preserved by reduction rules. Moreover, we show that continuous soundness is coNP-complete, and relates to integer boundedness.

We say that a workflow net $\mathcal{N}$ is *continuously sound* if for all continuous markings $\boldsymbol{m} \in \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \{\mathsf{i}\colon 1\})$ it is the case that $\boldsymbol{m} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$.

Theorem 1. *Continuous unsoundness implies generalised unsoundness.*

Proof. Let $\mathcal{N} = (P, T, F)$ be a workflow net that is not continuously sound. By definition of continuous soundness, there exists some continuous marking $\boldsymbol{m} \in \mathbb{Q}^P_{\geq 0}$ such that $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}$ and $\boldsymbol{m} \not\rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$. By Lemma 1, there exists $b \in \mathbb{N}_{\geq 1}$ such that $\{\mathsf{i}\colon b\} \rightarrow^* b \cdot \boldsymbol{m}$. Furthermore, by Lemma 1, $b \cdot \boldsymbol{m} \not\rightarrow^* \{\mathsf{f}\colon b\}$. This means that $\mathcal{N}$ is not b-sound, and consequently not generalised sound. ☐

Proposition 6. *The reduction rules from [10] preserve continuous soundness.*

Theorem 2. *Continuous soundness is coNP-complete. Moreover, coNP-hardness holds even if the underlying graph of the given workflow net is acyclic.*

Proof (of membership in coNP). The *inclusion problem* consists in determining whether, given Petri nets $\mathcal{N}$ and $\mathcal{N}'$ over a common set of places, and markings $\boldsymbol{m}$ and $\boldsymbol{m}'$, it is the case that $\mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \boldsymbol{m}) \subseteq \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}', \boldsymbol{m}')$. The inclusion problem is known to be coNP-complete [8, Prop. 4.6].

Let $\mathcal{N} = (P, T)$ be a workflow net. Let $\mathcal{N}^{-1} = (P, T^{-1})$ be defined as $\mathcal{N}$ but with its transitions reversed, *i.e.* where $T^{-1} := \{t^{-1} \mid t \in T\}$ with ${}^\bullet(t^{-1}) := t^\bullet$ and $(t^{-1})^\bullet := {}^\bullet t$. It is the case that $\boldsymbol{m} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}'$ in $\mathcal{N}$ iff $\boldsymbol{m}' \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}$ in $\mathcal{N}^{-1}$. Observe that $\mathcal{N}$ is continuously sound iff the following holds for all $\boldsymbol{m}$:

$$\boldsymbol{m} \in \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \{\mathsf{i}\colon 1\}) \implies \{\mathsf{f}\colon 1\} \in \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \boldsymbol{m}).$$

So, as $\{\mathsf{f}\colon 1\} \in \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \boldsymbol{m})$ is equivalent to $\boldsymbol{m} \in \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}^{-1}, \{\mathsf{f}\colon 1\})$, continuous soundness holds iff $\mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}, \{\mathsf{i}\colon 1\}) \subseteq \mathbb{Q}_{\geq 0}\text{-Reach}(\mathcal{N}^{-1}, \{\mathsf{f}\colon 1\})$. As inclusion can be tested in coNP, membership follows. ☐

Proof (of coNP-hardness). We give a reduction from the problem of determining whether a Boolean formula in disjunctive normal form (DNF) is a tautology. We adapt a construction from [32] used to show that soundness in acyclic workflow nets is coNP-hard. The proof is more challenging under the continuous semantics as several variable valuations and clauses can be simultaneously used.

The reduction is depicted in Figure 2 for $\varphi = (x_1 \wedge x_2 \wedge \neg x_4) \vee (\neg x_1 \wedge x_3 \wedge x_4)$. In general, let $\varphi = \bigvee_{j \in [1..k]} C_j$ be a Boolean formula in DNF with k clauses over variables $x_1, \ldots, x_m$. We define a workflow net $\mathcal{N}_\varphi = (P, T, F)$.

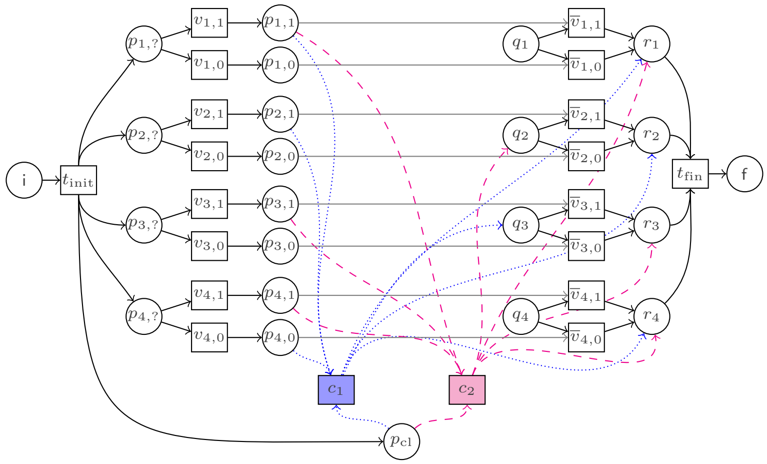

Fig. 2. A workflow net $\mathcal{N}_\varphi$ such that $\mathcal{N}_\varphi$ is continuously sound iff $\varphi = (x_1 \land x_2 \land \neg x_4) \lor (x_1 \land x_3 \land x_4)$ is a tautology. Places and transitions contain their names (not values). Arcs corresponding to the first and second clauses are respectively dotted and dashed.

Definition. The places are defined as $P \coloneqq \{\mathsf{i}, p_{\text{cl}}, \mathsf{f}\} \cup P_{\text{var}} \cup P_{\text{clean}}$, where $P_{\text{var}} \coloneqq \bigcup_{i \in [1..m]} \{p_{i,?}, p_{i,1}, p_{i,0}\}$ and $P_{\text{clean}} \coloneqq \bigcup_{i \in [1..m]} \{q_i, r_i\}$. The transitions are defined as $T \coloneqq \{t_{\text{init}}, t_{\text{fin}}\} \cup T_{\text{var}} \cup T_{\text{clauses}} \cup T_{\overline{\text{var}}}$, where

$$T_{\text{var}} \coloneqq \bigcup_{i \in [1..m]} \{v_{i,1}, v_{i,0}\}, T_{\text{clauses}} \coloneqq \{c_i \mid i \in [1..k]\} \text{ and } T_{\overline{\text{var}}} \coloneqq \bigcup_{i \in [1..m]} \{\overline{v}_{i,1}, \overline{v}_{i,0}\}.$$

Let us explain how $\mathcal{N}_\varphi$ is *intended* to work. Transition t_{init} enables the initialization of variables and the selection of a clause that satisfies φ, i.e. ${}^\bullet t_{\text{init}} \coloneqq \{\mathsf{i}\colon 1\}$ and $t_{\text{init}}^\bullet \coloneqq \{p_{i,?}\colon 1 \mid i \in [1..m]\} + \{p_{\text{cl}}\colon 1\}$. A token in place $p_{i,b}$ indicates that variable x_i has been assigned value b (where "?" indicates "none"). Consequently, we have ${}^\bullet v_{i,b} \coloneqq p_{i,?}$ and $v_{i,b}^\bullet \coloneqq p_{i,b}$ for each $i \in [1..m]$ and $b \in \{0,1\}$.

Transition c_j consumes a token associated to each literal of clause C_j, *i.e.* ${}^\bullet c_j \coloneqq \{v_{i,1} \mid x_i \in C_j\} + \{v_{i,0} \mid \neg x_i \in C_j\}$. A token in place q_i indicates that variable x_i is not needed anymore (due to some satisfied clause). A token in place r_i indicates that variable x_i has been discarded. Therefore, transition c_j produces these tokens: $c_j^\bullet \coloneqq \{q_i \mid x_i \notin C_j \land \neg x_i \notin C_j\} + \{r_i \mid x_i \in C_j \lor \neg x_i \in C_j\}$.

Transition $\overline{v}_{i,b}$ discards variable x_i, i.e. ${}^\bullet \overline{v}_{i,b} \coloneqq \{p_{i,b}, q_i\}$ and ${}^\bullet \overline{v}_{i,b} \coloneqq \{q_i\}$. Once each variable is discarded, transition t_{fin} terminates the execution, i.e. ${}^\bullet t_{\text{fin}} \coloneqq \{r_i \mid i \in [1..m]\}$ and $t_{\text{fin}}^\bullet \coloneqq \{\mathsf{f}\colon 1\}$.

Correctness. Note that under $\rightarrow_{\mathbb{Q}_{\geq 0}}^*$, the workflow net needs not to proceed as described. Indeed, it could, e.g., assign half a token to $p_{i,0}$ and half a token to $p_{i,1}$. Similarly, several clauses can be used, with distinct scaling factors. Nonetheless, $\mathcal{N}_\varphi$ is continuously sound iff φ is a tautology.

$\Rightarrow$) Let $b_1, \ldots, b_m \in \{0,1\}$. Let $\pi \coloneqq t_{\text{init}} v_{1,b_1} \cdots v_{m,b_m}$. We have: $\{\mathsf{i}\colon 1\} \rightarrow_{\mathbb{Q}_{\geq 0}}^\pi \{v_{i,b_i}\colon 1 \mid i \in [1..m]\} + \{p_{\text{cl}}\colon 1\}$. Since $\mathcal{N}_\varphi$ is continuously sound by assumption, there must exists some $j \in [1..k]$ such that c_j is enabled. This implies that clause C_j is satisfied by the assignment. Hence, φ is a tautology.

$\Leftarrow$) The proof is technical and involves several invariants (see appendix). □

We may now prove that any nonredundant workflow net that is integer unbounded is also continuously unsound (the reverse is not necessarily true). Therefore, integer unboundedness relates to continuous soundness much like continuous unsoundness relates to generalised soundness.

Proposition 7. *Let $\mathcal{N}$ be a nonredundant workflow net and $\boldsymbol{m} \in \mathbb{N}^P$. If $(\mathcal{N}, \boldsymbol{m})$ is integer unbounded, then $\mathcal{N}$ is not continuously sound.*

Proof. Let $\mathcal{N} = (P, T, F)$ and $\boldsymbol{m} \in \mathbb{N}^P$ be such that $(\mathcal{N}, \boldsymbol{m})$ is not integer bounded. By Proposition 4, there exists $\boldsymbol{m}' > \boldsymbol{0}$ such that $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{m}'$. By nonredundancy, there exist $\lambda \in \mathbb{N}_{\geq 1}$ and $\boldsymbol{m}'' \in \mathbb{N}^P$ such that $\{\mathsf{i}\colon \lambda\} \to^* \{\mathsf{f}\colon 1\} + \boldsymbol{m}''$.

In [22, Lemma 12], it is shown that $\{\mathsf{i}\colon k\} \to_{\mathbb{Z}}^* \boldsymbol{n}$ implies the existence of some $\ell \in \mathbb{N}$ such that $\{\mathsf{i}\colon k + \ell\} \to^* \{\mathsf{f}\colon \ell\} + \boldsymbol{n}$. By invoking this lemma with $k \coloneqq 0$ and $\boldsymbol{n} \coloneqq \boldsymbol{m}'$, we obtain $\{\mathsf{i}\colon \ell\} \to^* \{\mathsf{f}\colon \ell\} + \boldsymbol{m}'$ for some $\ell \in \mathbb{N}$.

Altogether, $\{\mathsf{i}\colon \lambda + \ell\} \to^* \{\mathsf{f}\colon \lambda + \ell\} + \boldsymbol{m}' + \boldsymbol{m}''$. Since $\lambda + \ell \geq 1$, Lemma 1 yields $\{\mathsf{i}\colon 1\} \to_{\mathbb{Q}_{\geq 0}}^* \{\mathsf{f}\colon 1\} + \boldsymbol{m}'''$ where $\boldsymbol{m}''' \coloneqq (1/(\lambda + \ell))\boldsymbol{m}'$. As every transition of a workflow net produces at least one token, this contradicts the fact that $\mathcal{N}$ is continuously sound. Indeed, it is impossible to fully get rid of $\boldsymbol{m}''' > \boldsymbol{0}$. □

5 Using relaxations for structural soundness

A workflow net $\mathcal{N}$ is *k-quasi-sound* if $\{\mathsf{i}\colon k\} \to^* \{\mathsf{f}\colon k\}$. Furthermore, $\mathcal{N}$ is *structurally quasi-sound* if it is k-quasi-sound for some $k \in \mathbb{N}_{\geq 1}$.

As observed in [33], structural quasi-soundness is a necessary condition for structural soundness. The notion of structural quasi-soundness is naturally generalised to an arbitrary Petri net $\mathcal{N} = (P, T, F)$. Given markings $\boldsymbol{m}, \boldsymbol{m}' \in \mathbb{N}^P$, we say that $\boldsymbol{m}$ *structurally reaches* $\boldsymbol{m}'$ in $\mathcal{N}$ if $k \cdot \boldsymbol{m} \to^* k \cdot \boldsymbol{m}'$ for some $k \in \mathbb{N}_{\geq 1}$. A workflow net is structurally quasi-sound iff $\boldsymbol{m} \coloneqq \{\mathsf{i}\colon 1\}$ structurally reaches $\boldsymbol{m}' \coloneqq \{\mathsf{f}\colon 1\}$. So, the observation of [33] can be rephrased as follows.

Proposition 8. *Let $\mathcal{N}$ be a workflow net. If $\{\mathsf{i}\colon 1\}$ does not structurally reach $\{\mathsf{f}\colon 1\}$ in $\mathcal{N}$, then $\mathcal{N}$ is not structurally sound.*

The problem of structural quasi-soundness can be reduced to an instance of the Petri net reachability problem [33, Lemma 2.1]. Intuitively, the reduction produces a Petri net that nondeterministically chooses multiples of $\{\mathsf{i}\colon 1\}$ and $\{\mathsf{f}\colon 1\}$ for which to check reachability. Such an approach has a prohibitive computational cost as Petri net reachability is Ackermann-complete. However, we observe that structural reachability, and hence structural quasi-soundness, is equivalent to continuous reachability by Lemma 1.

Proposition 9. *Let $\mathcal{N} = (P, T, F)$ be a Petri net, and let $\boldsymbol{m}, \boldsymbol{m}' \in \mathbb{N}^P$ be markings. It is the case that $\boldsymbol{m}$ structurally reaches $\boldsymbol{m}'$ iff $\boldsymbol{m} \to_{\mathbb{Q}_{\geq 0}}^* \boldsymbol{m}'$.*

For a workflow net $\mathcal{N} = (P, T, F)$, let $k_\mathcal{N} \in \mathbb{N}_{\geq 1} \cup \{\infty\}$ be the smallest number for which $\mathcal{N}$ is $k_\mathcal{N}$-quasi-sound. Then $\mathcal{N}$ is structurally sound iff $k_\mathcal{N} \neq \infty$ and $\mathcal{N}$ is $k_\mathcal{N}$-sound [33, Thm 2.1]. By Proposition 9, $k_\mathcal{N} \neq \infty$ can be checked in polynomial time via a continuous reachability query. Moreover, a lower bound on $k_\mathcal{N}$ can be obtained by computing $k_{\mathcal{N},\mathbb{Z}} \in \mathbb{N}_{\geq 1} \cup \{\infty\}$, defined as the smallest value such that $\{\mathsf{i}\colon k\} \to_\mathbb{Z}^* \{\mathsf{f}\colon k\}$. We obtain a better bound by defining $k_{\mathcal{N},\mathbb{Q}_{\geq 0}} \in \mathbb{N}_{\geq 1} \cup \{\infty\}$ as the smallest value for which there is a continuous run $\pi = \lambda_1 t_1 \cdots \lambda_n t_n$ such that $\{\mathsf{i}\colon k\} \to_{\mathbb{Q}_{\geq 0}}^\pi \{\mathsf{f}\colon k\}$ and $\boldsymbol{\pi} \in \mathbb{N}^T$, where $\boldsymbol{\pi}[t] \coloneqq \sum_{i \in [1..n] : t_i = t} \lambda_i$. Values $k_{\mathcal{N},\mathbb{Z}}$ and $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ can respectively be computed by a translation to integer linear programming, and a decidable optimization modulo theory.

Proposition 10. *Let $\mathcal{N}$ be a workflow net. It is the case that $k_{\mathcal{N},\mathbb{Z}} \leq k_{\mathcal{N},\mathbb{Q}_{\geq 0}} \leq k_\mathcal{N}$. Moreover, $k_{\mathcal{N},\mathbb{Z}}$ can be computed from an integer linear program $\mathcal{P}$; $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ can be obtained by computing $\min k \in \mathbb{N}_{\geq 1} : \varphi(k)$ where φ is a formula from the existential fragment of mixed linear arithmetic φ, i.e. $\exists\mathsf{FO}(\mathbb{Q}, \mathbb{Z}, <, +)$; and both $\mathcal{P}$ and φ are constructible in polynomial time from $\mathcal{N}$.*

6 Free-choice workflow nets

Let $\mathcal{N} = (P, T, F)$ be a Petri net. We say that $\mathcal{N}$ is *free-choice* if for any $s, t \in T$, it is the case that either $\mathrm{supp}({}^\bullet s) \cap \mathrm{supp}({}^\bullet t) = \emptyset$ or ${}^\bullet s = {}^\bullet t$. For example, the nets $\mathcal{N}_\text{left}$ and $\mathcal{N}_\text{right}$ from Figure 1 are respectively free-choice and not free-choice.

It is known that generalised soundness is equivalent to 1-soundness in free-choice workflow nets [29]. We will show that the same holds for structural soundness, and that, surprisingly, for continuous soundness as well. This means that notions of soundness collapse for free-choice nets. This is proven in the forthcoming Lemma 2 and Theorem 3, which form one of the main theoretical contributions of this work.

Let $(\mathcal{N}, \boldsymbol{m})$ be a marked Petri net. We say that a transition t is *quasi-live* in $(\mathcal{N}, \boldsymbol{m})$ if there exists $\boldsymbol{m}'$ such that $\boldsymbol{m} \to^* \boldsymbol{m}' \to^t$. Similarly, we say that a transition t is *live* in $(\mathcal{N}, \boldsymbol{m})$ if for all $\boldsymbol{m}'$ such that $\boldsymbol{m} \to^* \boldsymbol{m}'$, t is quasi-live in $(\mathcal{N}, \boldsymbol{m}')$. In words, quasi-liveness states that there is at least one way to enable t, and liveness states that t can always be re-enabled. The set of *quasi-live* and *live* transitions of $(\mathcal{N}, \boldsymbol{m})$ are defined respectively as $F(m) \coloneqq \{t \in T \mid t \text{ is quasi-live in } (\mathcal{N}, \boldsymbol{m})\}$ and $L(m) \coloneqq \{t \in T \mid t \text{ is live in } (\mathcal{N}, \boldsymbol{m})\}$.

Lemma 2. *Let $\mathcal{N} = (P, T, F)$ be a free-choice Petri net, let $c \in \mathbb{N}_{\geq 1}$, and let $\boldsymbol{m} \in \mathbb{N}^P$. The following statements hold.*

1. *There exists a marking $\boldsymbol{m}'$ such that $\boldsymbol{m} \to^* \boldsymbol{m}'$ and $L(\boldsymbol{m}') = F(\boldsymbol{m}')$.*
2. *If $L(\boldsymbol{m}) = F(\boldsymbol{m})$, then $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m}) = F(\boldsymbol{m})$.*
3. *If $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m})$, $c \cdot \boldsymbol{m} \to^* \{\mathsf{f}\colon c\}$ and $(\mathcal{N}, c \cdot \boldsymbol{m})$ is bounded, then $\boldsymbol{m} = \{\mathsf{f}\colon 1\}$.*

Lemma 3. *Let $\mathcal{N}$ be a workflow net. If $\mathcal{N}$ is continuously sound, then $(\mathcal{N}, \{\mathsf{i}\colon k\})$ is bounded for all $k \in \mathbb{N}_{\geq 1}$.*

Theorem 3. *Let $\mathcal{N}$ be a free-choice workflow net. These statements are equivalent: (1) $\mathcal{N}$ is* 1*-sound, (2) $\mathcal{N}$ is generalised sound, (3) $\mathcal{N}$ is structurally sound, and (4) $\mathcal{N}$ is continuously sound.*

Proof. (1) $\Rightarrow$ (2). This was shown in [29].

(2) $\Rightarrow$ (3). By definition, if $\mathcal{N}$ is k-sound for all k, then it is for some k.

(2) $\Rightarrow$ (4). By Theorem 1.

(3) $\Rightarrow$ (1). Let $k \in \mathbb{N}_{\geq 1}$ be such that $\mathcal{N}$ is k-sound. Let $\boldsymbol{m} \in \mathbb{N}^P$ be such that $\{\mathsf{i}\colon 1\} \rightarrow^* \boldsymbol{m}$. By Lemma 2(1), there is a marking $\boldsymbol{m'} \in \mathbb{N}^P$ such that $\boldsymbol{m} \rightarrow^* \boldsymbol{m'}$ and $F(\boldsymbol{m'}) = L(\boldsymbol{m'})$. By Lemma 2(2), we have $L(k \cdot \boldsymbol{m'}) = F(k \cdot \boldsymbol{m'}) = F(\boldsymbol{m'})$.

By k-soundness, $(\mathcal{N}, \{\mathsf{i}\colon k\})$ must be bounded [9, Proposition 3.2 and Lemma 3.6]. Thus, since $\{\mathsf{i}\colon k\} \rightarrow^* k \cdot \boldsymbol{m} \rightarrow^* k \cdot \boldsymbol{m'}$, it is also the case that $(\mathcal{N}, k \cdot \boldsymbol{m'})$ is bounded. By k-soundness, $k \cdot \boldsymbol{m'} \rightarrow^* \{\mathsf{f}\colon k\}$. By invoking Lemma 2(3) with $c \coloneqq k$, we conclude that $\boldsymbol{m'} = \{\mathsf{f}\colon 1\}$. So, $\mathcal{N}$ is 1-sound as $\{\mathsf{i}\colon 1\} \rightarrow^* \boldsymbol{m} \rightarrow^* \boldsymbol{m'} = \{\mathsf{f}\colon 1\}$.

(4) $\Rightarrow$ (1). Assume that $\mathcal{N}$ is continuously sound. Let $\boldsymbol{m} \in \mathbb{N}^P$ be a marking such that $\{\mathsf{i}\colon 1\} \rightarrow^* \boldsymbol{m}$. By Lemma 2(1), there exists $\boldsymbol{m'} \in \mathbb{N}^P$ such that $\boldsymbol{m} \rightarrow^* \boldsymbol{m'}$ and $L(m') = F(m')$. Clearly, $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m'}$ and by continuous soundness $\boldsymbol{m'} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$. By Lemma 1, there exists $b \in \mathbb{N}_{\geq 1}$ such that $b \cdot \boldsymbol{m'} \rightarrow^* \{\mathsf{f}\colon b\}$.

By Lemma 3, continuous soundness of $\mathcal{N}$ implies that $(\mathcal{N}, b \cdot \boldsymbol{m'})$ is bounded, as $\{\mathsf{i}\colon b\} \rightarrow^* b \cdot \boldsymbol{m'}$. Since $L(m') = F(m')$, it follows from Lemma 2(2) that $L(b \cdot \boldsymbol{m'}) = F(b \cdot \boldsymbol{m'})$. By invoking Lemma 2(3) with $c \coloneqq b$, we derive $\boldsymbol{m'} = \{\mathsf{f}\colon 1\}$. Therefore, $\mathcal{N}$ is 1-sound as $\{\mathsf{i}\colon 1\} \rightarrow^* \boldsymbol{m} \rightarrow^* \boldsymbol{m'} = \{\mathsf{f}\colon 1\}$. ☐

7 Experimental evaluation

We implemented our approaches for generalised and structural soundness in C#.[4] We test continuous soundness via SMT solving. More precisely, we use an existential $\psi_{\mathcal{N}}$ formula of linear arithmetic, i.e. $\mathsf{FO}(\mathbb{Q}, <, +)$, from [8]. This formula is such that $\psi(\boldsymbol{m}, \boldsymbol{m'})$ holds iff $\boldsymbol{m} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m'}$ in $\mathcal{N}$. Continuous soundness amounts to the $\exists\forall$-formula $\psi_{\mathcal{N}}(\{\mathsf{i}\colon 1\}, \boldsymbol{m}) \wedge \neg\psi_{\mathcal{N}}(\boldsymbol{m}, \{\mathsf{f}\colon 1\})$. To solve such formulas, we use Z3 [27]. We further use Z3 to decide structural quasi-soundness and compute $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ (see Proposition 10), again via the formulas of [8].

We evaluated our prototype implementation on a standard benchmark suite used regularly in the literature, and a novel suite of synthetic instances where generalised or structural soundness are hard to decide with a naive approach.

We compared with two established tools for soundness: LoLA (v2.0) [37], and Woflan [35].[5] The latter can only decide *classical* soundness (1-soundness + quasi-liveness). Nonetheless, we use quasi-live instances, so for which 1-soundness and classical soundness are equivalent. We further use a transformation to reduce the verification of k-soundness to the one of 1-soundness [9, Lemma 3.6]. On the

[4] The implementation can be obtained from `https://doi.org/10.6084/m9.figshare.19721674.v2`.

[5] A version of Woflan suitable for running without user interaction was provided, via personal communication, by its maintainer.

other hand, LoLA can directly decide k-soundness. To do so, we start from $\{\mathsf{i} \colon k\}$ and check a CTL formula of the form $\forall \mathsf{G}\, \exists \mathsf{F}\, ((\boldsymbol{m}[\mathsf{f}] = k) \wedge \bigwedge_{p \neq \mathsf{f}} \boldsymbol{m}[p] = 0)$.

Experiments were run on an 8-Core Intel® Core™ i7-7700 CPU @ 3.60GHz with Ubuntu 18.04. We limited memory to ∼8GB, and time to 120s for each instance. Tools were called from a Python script. For LoLA and our implementation, we used the *time* module to measure time. Running Woflan involves some overhead, so we instead take the total verification time reported by Woflan itself.

7.1 Free-choice benchmark suite

The benchmark suite encompasses 1386 free-choice Petri nets that represent business processes modeled in the IBM WebSphere Business Modeler. It was originally presented in [17], and has been studied frequently in the literature [10,18]. These nets are not workflow nets by our definition, but can be transformed using a known procedure [24]. Intuitively, the nets are workflow nets with multiple final places, and the procedure adds a dedicated output place and ensures that the resulting workflow net represents the desired behaviour. However, roughly 1% of the nets are not workflow nets by our definition even after the procedure, as they contain nodes that are not on a path from i to f. We removed these nets.

We further checked each net for safety using LoLA and dropped unsafe nets. Recall that $(\mathcal{N}, \{\mathsf{i} \colon 1\})$ is sound if each reachable marking has at most one token per place. Unsafe instances can be dropped as unsafety implies 1-unsoundness in free-choice nets [36, Thm. 4.2 and 4.4], and as existing methods for checking safety, *e.g.* via state-space exploration with partial order reductions, are very efficient (here needing a mean of 3ms). Thus, we considered safe instances only. Among the 1386 instances, 1382 are workflow nets, and 977 are further safe.

We also invoked an implementation of the reduction rules of [10] to reduce the size of all instances.[6] As discussed in the introduction, the rules can reduce some instances to trivially sound nets. However, even the size of nontrivial reduced instances tends to be small, with an average number of places and transitions of roughly 14, while three quarters of nets have at most 18 places and transitions. This is small enough that a complete state-splace enumeration is often feasible, in particular as the nets are safe and especially LoLA utilizes powerful partial order reductions for such nets. As we want to focus on scalability, we chained instances to produce challenging synthetic nets based on real-world instances. This is a natural way of constructing workflow nets, intuitively, the final process can be composed of many subtasks. It can be seen as a special case of refinement operations, studied in the context of generalised soundness [21].

The chaining procedure merges two workflow nets $\mathcal{N} = (P, T, F)$ and $\mathcal{N}' = (P', T', F')$ into $\mathcal{N}'' \coloneqq (P'', T'', F'')$ where $P'' \coloneqq P \cup P'$, $T'' \coloneqq T \cup T' \cup \{t_{aux}\}$ with F'' as $F' + F''$ extended with ${}^\bullet t_{\text{aux}}[\mathsf{f}] \coloneqq 1$, $t_{\text{aux}}^\bullet[\mathsf{i}'] \coloneqq 1$, and ${}^\bullet t_{\text{aux}}[p] = t_{\text{aux}}^\bullet[p'] \coloneqq 0$ for other entries. It is readily seen that this construction (1) produces a free-choice net if both $\mathcal{N}$ and $\mathcal{N}'$ are free-choice; and (2) preserves safety.

[6] At time of writing, an implementation is available at https://github.com/LoW12/Hadara-AdSimul.

This way, we generated large instances by using $\ell \in \{1, 21, 41, \ldots, 401\}$ randomly chosen unreduced safe instances from the benchmark suite as inputs to be chained into one instance, then reduced that instance. For each number ℓ, we produced 20 combined nets, with a fresh random choice each time, in order to have a more representative collection of nets for ℓ. This resulted in 420 instances, of which 405 are nontrivial after applying reduction rules.

A caveat is that such large nets may seem unlikely to arise in practice. It seems a human designer would avoid designing highly complex processes corresponding to Petri nets with thousands of places. However, process models are not only explicitly written by humans, but also machine-generated, *e.g.* by mining event logs (see [34] for a book on the topic). In particular, being free-choice is preserved by chaining, so a large free-choice net may "hide" and combine several less complex processes, which might necessitate analyzing large workflow nets.

Results. We checked the safe free-choice instances obtained as explained above for 1-soundness using LoLA, Woflan and our implementation of continuous soundness. The results are shown on the left of Figure 3. The right-hand side of the figure provides an overview over the sizes of the nets. In each case, N refers to the number of original instances that were chained to create each instance.

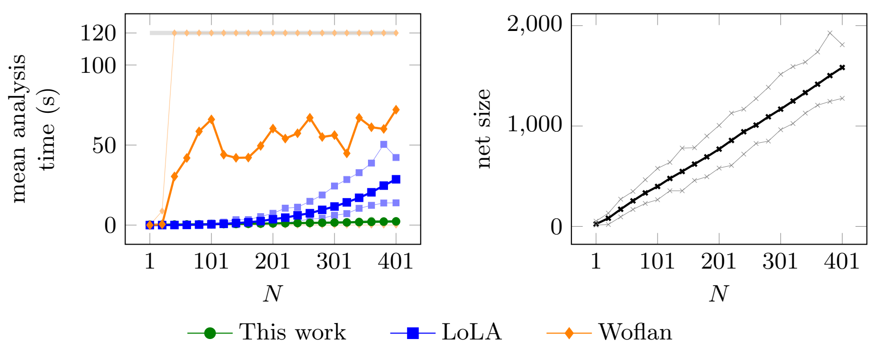

Fig. 3. Experiments on chained free-choice instances. The x-value denotes the number N of chained nets. Dark thick lines denote the mean, and light thin lines of the same color denote the minimum and maximum, respectively. For Woflan, the minimum line is slightly below the line of this work. For this work, the minimum and maximum lines are very close to the mean. *Left:* The y-value denotes time for checking soundness of the 20 nets for each N. Marks on the gray line at 120s denote timeouts. *Right:* The y-value denotes the size of generated nets.

The results show that state-space exploration via LoLA is very fast for moderate sizes, but does not scale as well. Continuous soundness is in fact outperformed by LoLA for $N \leq 100$, but scales much better, showing essentially linear growth in the given data range. For instance, continuous soundness takes a mean of 0.25s for $N = 1$, a mean of 1.07s for $N = 201$, and a mean of 2.28s for $N = 401$.

Woflan performs very well on the original instances, but times out frequently for larger instances. Woflan checks so-called S-coverability [36]. This is fast on many instances, even large ones, but starts running into the exponential-time worst case when instances get larger. For $N = 1$ and $N = 21$, Woflan does not ever time out, while it times out for roughly half of the instances in the range from $N = 201$ to $N = 401$. Overall, we infer that for large free-choice workflow nets, deciding soundness by checking continuous soundness can outperform existing techniques, while the procedure is still competitive on moderate instances.

7.2 Synthetic instances

In the previously discussed benchmark suite, nets are free-choice. So structural and generalised soundness are equivalent by Theorem 3. We considered including a second suite of 590 non-free-choice Petri nets that represent processes of the SAP reference model [26]. However them turn out to be 1-quasi-sound but not 1-sound, so they represent trivial cases for generalised and structural soundness: simply checking 1-soundness, or 1-quasi-soundness and then 1-soundness, decides all instances. It's also worth mentioning that none of the 590 SAP instances are continuously sound, so all of them can be shown to not be generalised sound by checking continuous soundness, without having to check 1-soundness.

In order to have a wider variety of challenging instances, we introduce several families of synthetic workflow nets. The nets are simple to understand, but have large numbers of reachable marking, so are challenging for approaches relying on state-space exploration, *e.g.* model checking.

Encoding arc weights. To simplify the presentation, we describe synthetic instances utilizing arcs with weights. For benchmarking, we removed the arc weights and instead input equivalent weightless nets. To do so, we used an encoding that simulates exponentially large weights by polynomially many transitions and places (the encoding is explained in Appendix A.5). It preserves (quasi-)soundness, but significantly increases the number of reachable markings. Indeed, our synthetic instances are mostly trivial to solve by enumerating reachable markings when arcs have weights, but become much harder to decide when the encoding is used.[7] While much of the literature on workflow nets does not consider nets with arc weights, implicit structural encodings can occur in practice.

Generalised soundness

Benchmark instances. We introduce a synthetic family of nets where generalised soundness appears to be challenging. The family $\{\mathcal{N}_c\}_{c \in \mathbb{N}_{\geq 1}}$ is defined at the top of Figure 4. Parameter $c \in \mathbb{N}_{\geq 1}$ is the smallest value for which $\mathcal{N}_c$ is c-unsound. From $\{\mathsf{i} \colon c\}$, the sequence $t_{\mathsf{i}}^c t_r^{c+1}$ can be fired, which leads to the deadlock $\{r \colon c+$

[7] It is deliberately used to make instances challenging, not to ensure compatibility with LoLA or Woflan, as both support arc weights.

1}. Yet, when starting with $k < c$ tokens in i, and firing t_{i}^k, transitions t_r and t_{f} can only be fired exactly k times, and $\{\mathsf{f}\colon k\}$ will be reached.

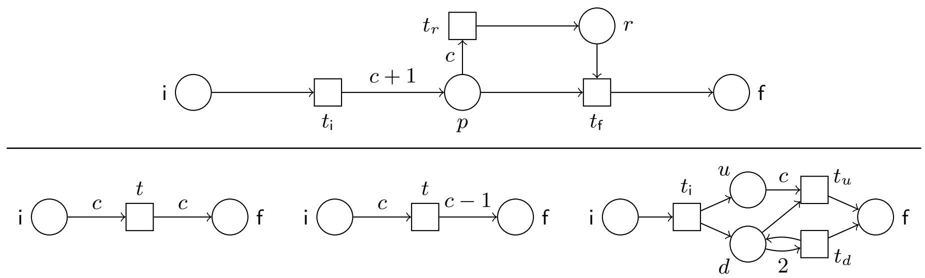

Fig. 4. *Top*: A workflow net $\mathcal{N}_c$ that is c-unsound and k-sound for all $k \in [1..c-1]$. *Bottom*: Three families of instances. *Bottom left:* $\mathcal{N}_{\text{sound-}c}$ is quasi-sound and ℓc-sound for all $\ell \in \mathbb{N}_{\geq 1}$. *Bottom center:* $\mathcal{N}_{\neg\text{quasi-}c}$ is not structurally quasi-sound. *Bottom right:* $\mathcal{N}_{\neg\text{sound-}c}$ is ℓc-quasi-sound for all $\ell \in \mathbb{N}_{\geq 1}$, but not structurally sound.

The naive approach to decide generalised soundness is to check k-soundness for all k until a counterexample is found or a bound is exceeded. It is known that if a counterexample exists, then there also is one of size at most exponential [9, Lemma 5.6 and 5.8]. The approach we chose for semi-deciding generalised soundness is to check continuous soundness. Recall that continuous soundness is a necessary (albeit not sufficient) condition, as shown in Theorem 1.

In our evaluation, we used Woflan and LoLA to check generalised soundness of the family for different c by checking 1-sound, ..., c-soundness, and compared the result to the time needed for testing continuous soundness. Our main goal is to evaluate whether checking continuous soundness is efficient enough to serve as an inexpensive way to witness generalised unsoundness for nontrivial instances.

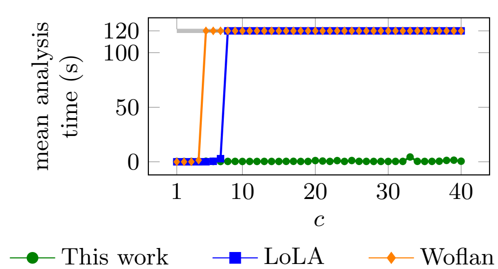

Fig. 5. Time to check generalised soundness of $\mathcal{N}_c$ for different values of c. Marks on the gray line at $120s$ denote timeouts.

Results. Figure 5 depicts the results. Woflan and LoLA show good performance for small values of c, but do not scale well to larger values. They respectively time out for $c \geq 5$ and $c \geq 8$. The instances are not free-choice, so LoLA and Woflan need to explore the state-space for each $k \leq c$, which becomes infeasible. For $c \geq 14$, Woflan cannot even check 1-soundness within the time limit. LoLA can check 1- and 2-soundness for $c \leq 28$, but cannot handle 2-soundness for larger c. Continuous soundness is efficiently verifiable even for $c = 40$. In particular, we need less than $5s$ on all instances. The greatest time is at $c = 33$. Further, at most $1s$ is needed on 34 out of 40 instances (mean of $0.6s$).

Structural soundness

Benchmark instances. For structural soundness, recall that our decision procedure is based on checking structural quasi-soundness and obtaining some lower bound for the smallest number for which the net is quasi-sound. Thus, we want to test on both benchmark instances that are structurally quasi-sound and those that are not. We introduce three families of non-free-choice nets for which structural soundness appears challenging. These instances are defined at the bottom of Figure 4. We respectively denote them $\mathcal{N}_{\text{sound-}c}$ (left), $\mathcal{N}_{\neg\text{quasi-}c}$ (center) and $\mathcal{N}_{\neg\text{sound-}c}$ (right). We claim that: $\mathcal{N}_{\text{sound-}c}$ is ℓc-sound for all $\ell \in \mathbb{N}_{\geq 1}$; $\mathcal{N}_{\neg\text{quasi-}c}$ is not structurally quasi-sound; $\mathcal{N}_{\neg\text{sound-}c}$ is ℓc-quasi-sound for all $\ell \in \mathbb{N}_{\geq 1}$, not k-quasi-sound for any other number $k \in \mathbb{N}_{\geq 1}$, and not structurally sound.

For the experiments, our goal is twofold. First, we want to evaluate whether utilizing continuous reachability to decide structural quasi-soundness is more efficient than using the known reduction to reachability described in [33, Lemma 2.1]. Woflan does not directly support checking reachability, so we only compare with LoLA. Second, we want to evaluate whether the lower bound for the smallest number for which the net is quasi-sound, which we dubbed $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ towards the end of Section 5, is close to the actual smallest number, dubbed $k_{\mathcal{N}}$.

A caveat of this evaluation is that we evaluate only on our synthetic instances, and that computing $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ is only one step in deciding structural soundness. However, we think that the evaluation on these hard synthetic instances can give insights into the applicability on nontrivial real-world instances.

Results. Figure 6 compares the time needed to verify structural reachability for LoLA and our prototype. For small instances, LoLA sometimes performs very well, but we scale better for large values. Of particular note is that in the absence of quasi-soundness, LoLA will generate an infinite state-space, so will generally run out of time or memory. In particular, LoLA times out for all c on $\mathbb{N}_{\neg\text{quasi-}c}$. It also times out for $c \geq 32$ on $\mathbb{N}_{\neg\text{sound-}c}$. On the other hand, continuous soundness never times out for the given values of c. In fact, when we tested continuous soundness for much larger values of c, we found that our implementation of continuous reachability decides structural quasi-soundness for $N_{\neg\text{quasi-}c}$ in under $2s$ for $c = 20\ 000\ 000$.

We further found that for all instances, $k_{\mathcal{N},\mathbb{Q}_{\geq 0}} = k_{\mathcal{N}}$, that is, our lower bound exactly matches the smallest number for which the net is quasi-sound.

Thus, it only remains to decide $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$-quasi-soundness and $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$-soundness in order to decide structural soundness. This is in contrast to the naive approach, which starts at $k = 1$ and checks k-quasi-soundness for each value up to $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$.

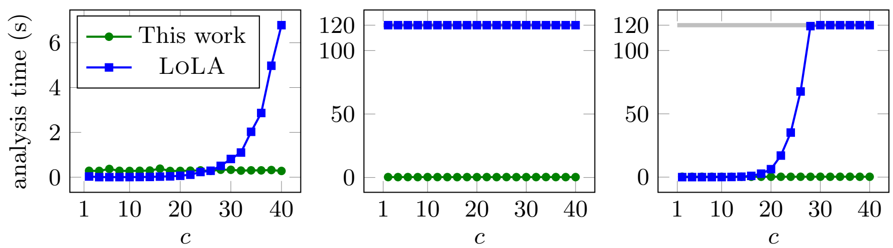

Fig. 6. Time taken vs parameter c for checking structural quasi-soundness using the reduction to reachability, and utilizing our approach to compute $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$, for each of the three families at the bottom of Figure 4: $\mathcal{N}_{\text{sound-}c}$ (*left*),$\mathcal{N}_{\neg\text{quasi-}c}$ (*center*), $\mathcal{N}_{\neg\text{sound-}c}$ (*right*). Note that the axis ranges differ. Marks on the gray line at 120s denote timeouts.

8 Conclusion

In this work, we have shown how reachability relaxations allow to efficiently semi-decide generalised and structural soundness. Our approach combines nicely with reduction rules, as they all preserve relaxations. In particular, we have introduced continuous soundness as an approximation of generalised soundness, and shown that it coincides with other types of soundness for free-choice nets.

As part of future work, we plan to migrate our prototype into the process mining framework ProM, to make the algorithms available to practitioners.

Acknowledgements

We thank Dirk Fahland and Eric Verbeek for their help with Woflan. Michael Blondin was supported by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada (NSERC), and by the Fonds de recherche du Qubec Nature et technologies (FRQNT).

References

1. van der Aalst, W.M.P.: Verification of workflow nets. In: Proc. 18th International Conference on Application and Theory of Petri Nets (ICATPN). vol. 1248, pp. 407–426 (1997). https://doi.org/10.1007/3-540-63139-9_48
2. van der Aalst, W.M.P.: The application of Petri nets to workflow management. Journal of Circuits, Systems, and Computers **8**(1), 21–66 (1998). https://doi.org/10.1142/S0218126698000043

3. van der Aalst, W.M.P.: Workflow verification: Finding control-flow errors using Petri-net-based techniques. In: Business Process Management, Models, Techniques, and Empirical Studies. pp. 161–183 (2000). https://doi.org/10.1007/3-540-45594-9_11
4. van der Aalst, W.M.P., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. Cooperative information systems, MIT Press (2002)
5. Athanasiou, K., Liu, P., Wahl, T.: Unbounded-thread program verification using thread-state equations. In: Proc. 8th International Joint Conference on Automated Reasoning (IJCAR). pp. 516–531 (2016). https://doi.org/10.1007/978-3-319-40229-1_35
6. Barkaoui, K., Petrucci, L.: Structural analysis of workflow nets with shared resources. In: Proc. Workflow Management: Net-based Concepts, Models, Techniques and Tools (WFM). vol. 98/7, pp. 82–95 (1998)
7. Blondin, M.: The ABCs of Petri net reachability relaxations. ACM SIGLOG News **7**(3) (2020). https://doi.org/10.1145/3436980.3436984
8. Blondin, M., Finkel, A., Haase, C., Haddad, S.: The logical view on continuous Petri nets. ACM Transactions on Computational Logic (TOCL) **18**(3), 24:1–24:28 (2017). https://doi.org/10.1145/3105908
9. Blondin, M., Mazowiecki, F., Offtermatt, P.: The complexity of soundness in workflow nets. In: Proc. 37th Symposium on Logic in Computer Science (LICS) (2022)
10. Bride, H., Kouchnarenko, O., Peureux, F.: Reduction of workflow nets for generalised soundness verification. In: Proc. 18th International Conference Verification, Model Checking, and Abstract Interpretation (VMCAI). pp. 91–111 (2017). https://doi.org/10.1007/978-3-319-52234-0_6
11. Cardoza, E., Lipton, R.J., Meyer, A.R.: Exponential space complete problems for Petri nets and commutative semigroups: Preliminary report. In: Proc. 8th Annual ACM Symposium on Theory of Computing (STOC). pp. 50–54 (1976). https://doi.org/10.1145/800113.803630
12. Chistikov, D., Haase, C., Halfon, S.: Context-free commutative grammars with integer counters and resets. Theoretical Computer Science **735**, 147–161 (2018). https://doi.org/10.1016/j.tcs.2016.06.017, `https://doi.org/10.1016/j.tcs.2016.06.017`
13. Czerwinski, W., Orlikowski, L.: Reachability in vector addition systems is Ackermann-complete. In: Proc. 62^{nd} Annual IEEE Symposium on Foundations of Computer Science (FOCS) (2021), to appear
14. Desel, J., Esparza, J.: Free Choice Petri Nets. Cambridge University Press (1995). https://doi.org/10.1017/CBO9780511526558
15. Desel, J., Esparza, J.: Free Choice Petri Nets. Cambridge University Press (1995). https://doi.org/10.1017/CBO9780511526558
16. Esparza, J., Ledesma-Garza, R., Majumdar, R., Meyer, P.J., Niki, F.: An SMT-based approach to coverability analysis. In: Proc. 26th International Conference on Computer Aided Verification (CAV). pp. 603–619 (2014). https://doi.org/10.1007/978-3-319-08867-9_40
17. Fahland, D., Favre, C., Jobstmann, B., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Instantaneous soundness checking of industrial business process models. In: Proc. 7th International Conference on Business Process Management (BPM). pp. 278–293 (2009). https://doi.org/10.1007/978-3-642-03848-8_19
18. Favre, C., Völzer, H., Müller, P.: Diagnostic information for control-flow analysis of workflow graphs (a.k.a. free-choice workflow nets). In: Proc. 22nd Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS). pp. 463–479 (2016). https://doi.org/10.1007/978-3-662-49674-9_27

19. Fraca, E., Haddad, S.: Complexity analysis of continuous Petri nets. Fundamenta Informaticae **137**(1), 1–28 (2015). https://doi.org/10.3233/FI-2015-1168
20. van Hee, K.M., Oanea, O., Sidorova, N., Voorhoeve, M.: Verifying generalized soundness of workflow nets. In: Proc. 6th International Andrei Ershov Memorial Conference on Perspectives of Systems Informatics (PSI). pp. 235–247 (2006). https://doi.org/10.1007/978-3-540-70881-0_21
21. van Hee, K.M., Sidorova, N., Voorhoeve, M.: Soundness and separability of workflow nets in the stepwise refinement approach. In: Proc. 24th International Conference on Applications and Theory of Petri Nets 2003 (ICATPN). vol. 2679, pp. 337–356 (2003). https://doi.org/10.1007/3-540-44919-1_22
22. van Hee, K.M., Sidorova, N., Voorhoeve, M.: Generalised soundness of workflow nets is decidable. In: Proc. 25th International Conference on Applications and Theory of Petri Nets (ICATPN). pp. 197–215 (2004). https://doi.org/10.1007/978-3-540-27793-4_12
23. Hoffmann, P.E.: Workflow Nets: Reduction Rules and Games. Ph.D. thesis, Technische Universität München (2017)
24. Kiepuszewski, B., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Fundamentals of control flow in workflows. Acta Informatica **39**(3), 143–209 (2003). https://doi.org/10.1007/s00236-002-0105-4
25. Leroux, J.: The reachability problem for Petri nets is not primitive recursive. In: Proc. 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS) (2021), to appear
26. Mendling, J., Moser, M., Neumann, G., Verbeek, H.M.W., van Dongen, B.F., van der Aalst, W.M.P.: Faulty EPCs in the SAP reference model. In: Proc. 4th International Conference on Business Process Management (BPM). pp. 451–457 (2006). https://doi.org/10.1007/11841760_38
27. de Moura, L.M., Bjørner, N.: Z3: an efficient SMT solver. In: Proc. 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS). pp. 337–340. Springer (2008). https://doi.org/10.1007/978-3-540-78800-3_24, tool available at `https://github.com/Z3Prover/z3`.
28. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE **77**(4), 541–580 (1989). https://doi.org/10.1109/5.24143
29. Ping, L., Hao, H., Jian, L.: On 1-soundness and soundness of workflow nets. In: Third Workshop on Modelling of Objects, Components, and Agents. p. 21 (2004)
30. Rackoff, C.: The covering and boundedness problems for vector addition systems. Theoretical Computer Science **6**, 223–231 (1978). https://doi.org/10.1016/0304-3975(78)90036-1
31. Schrijver, A.: Theory of Linear and Integer Programming. John Wiley & Sons (1986)
32. Ţiplea, F.L., Bocăneală, C., Chiroşcă, R.: On the complexity of deciding soundness of acyclic workflow nets. IEEE Transactions on Systems, Man, and Cybernetics: Systems **45**(9), 1292–1298 (2015). https://doi.org/10.1109/TSMC.2015.2394735
33. Ţiplea, F.L., Marinescu, D.C.: Structural soundness of workflow nets is decidable. Inf. Process. Lett. **96**(2), 54–58 (2005). https://doi.org/10.1016/j.ipl.2005.06.002
34. Van Der Aalst, W.: Data science in action. In: Process mining, pp. 3–23. Springer, 2 edn. (2016)
35. Verbeek, E., van der Aalst, W.M.P.: Woflan 2.0: A Petri-net-based workflow diagnosis tool. In: Proc. 21st Conference on Application and Theory of Petri Nets 2000, , (ICATPN). pp. 475–484 (2000). https://doi.org/10.1007/3-540-44988-4_28

36. Verbeek, H.M.W., Basten, T., van der Aalst, W.M.P.: Diagnosing workflow processes using woflan. The Computer Journal **44**(4), 246–279 (2001). https://doi.org/10.1093/comjnl/44.4.246
37. Wolf, K.: Petri net model checking with LoLA 2. In: Application and Theory of Petri Nets and Concurrency. pp. 351–362 (2018)

A Appendix

A.1 Missing proofs of Section 3

Lemma 1. *Let $\boldsymbol{m}$, $\boldsymbol{m'}$ be continuous markings. It is the case that $\boldsymbol{m} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m'}$ iff there exists $b \in \mathbb{N}_{\geq 1}$ such that $b \cdot \boldsymbol{m} \rightarrow^* b \cdot \boldsymbol{m'}$.*

Proof. $\Leftarrow$) Let $b \cdot m \rightarrow^{\pi} b \cdot \boldsymbol{m'}$. Let $\beta \coloneqq 1/b$. Let us prove that $\boldsymbol{m} \rightarrow^{\beta \cdot \pi} \boldsymbol{m'}$. To do so, we show that $b \cdot \boldsymbol{m} \rightarrow^{\pi[1:n]} \boldsymbol{m}_n$ implies $\boldsymbol{m} \rightarrow^{\pi[1:n]}_{\mathbb{Q}_{\geq 0}} \beta \cdot \boldsymbol{m}_n$. Let us proceed by induction on n. Assume that

$$b \cdot m \rightarrow^{\pi[1:n]} \boldsymbol{m}_n \rightarrow^{t_{n+1}} \boldsymbol{m}_{n+1} \text{ where } t_{n+1} \coloneqq \pi[n+1].$$

By induction hypothesis, we have $\boldsymbol{m} \rightarrow^{\pi[1:n]}_{\mathbb{Q}_{\geq 0}} \beta \cdot \boldsymbol{m}_n$. Note that $\boldsymbol{m}_{n+1} = \boldsymbol{m}_n + \Delta(t_{n+1})$. So, $\beta \cdot \boldsymbol{m}_n + \beta \cdot \Delta(t_{n+1}) = \beta \cdot \boldsymbol{m}_{n+1}$. Thus, $\beta \cdot t_{n+1}$ has the right effect to lead from $\boldsymbol{m}_n$ to $\boldsymbol{m}_{n+1}$. It only remains to show that $\beta \cdot t_{n+1}$ is enabled at $\boldsymbol{m}_n$. Note that t_{n+1} is enabled at $\boldsymbol{m}_n$, hence by definition, $\boldsymbol{m}_n[p] \geq {}^\bullet t_{n+1}[p]$ for all $p \in P$. It follows that $\beta \cdot \boldsymbol{m}_n[p] > \beta \cdot {}^\bullet t_{n+1}[p]$, so $\beta \cdot t_{n+1}$ is enabled in $\boldsymbol{m}_n$.

$\Rightarrow$) Let $\boldsymbol{m} \rightarrow^{\pi}_{\mathbb{Q}_{\geq 0}} \boldsymbol{m'}$. Let β be the product of the scaling factors denominators along π. Let us show that $b \cdot \boldsymbol{m} \rightarrow^{\pi'} b \cdot \boldsymbol{m'}$. We establish the following for all n:

$$\text{if } \boldsymbol{m} \rightarrow^{\pi[1:n]}_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}_n, \text{ then there exists } \pi'_n \text{ such that } b \cdot \boldsymbol{m} \rightarrow^{\pi'_n} b \cdot \boldsymbol{m}_n.$$

Assume this holds for some n. Let $\alpha \cdot t_{n+1} = \pi[n]$. We show the following:

$$\text{if } \boldsymbol{m}_n \rightarrow^{\alpha t_{n+1}}_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}_{n+1}, \text{then } b \cdot \boldsymbol{m}_n \rightarrow^{(t_{n+1})^{b \cdot \alpha}} b \cdot \boldsymbol{m}_{n+1}.$$

First, let us argue that $b \cdot \alpha$ is an integer. Note that by the fact that scaling factors are chosen from $(0, 1]$, it follows that α can be written as u/d for some $u, d \in \mathbb{N}$ where $d \neq 0$. Further, note that b was chosen as the product of all denominators of scaling factors along π. In particular, d is a factor of b, so we have $b = d \cdot b'$ for some $b' \in \mathbb{N}$, and thus $b \cdot \alpha = d \cdot b' \cdot u/d = b' \cdot u$. Next, let us argue that $(t_{n+1})^{b \cdot \alpha}$ has the right effect to lead from $b \cdot \boldsymbol{m}_n$ to $b \cdot \boldsymbol{m}_{n+1}$. Note that $\boldsymbol{m}_{n+1} = \boldsymbol{m}_n + \alpha \cdot \Delta(t_{n+1})$. So, $b \cdot \boldsymbol{m}_{n+1} = b \cdot \boldsymbol{m}_n + b \cdot \alpha \Delta(t_{n+1}) = b \cdot \boldsymbol{m}_n + \Delta((t_{n+1})^{b \cdot \alpha})$. It remains to argue that $(t_{n+1})^{b \cdot \alpha}$ is fireable from $b \cdot \boldsymbol{m}_n$. By $\boldsymbol{m}_n \rightarrow^{\alpha t_{n+1}}_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}_{n+1}$, it follows that $\boldsymbol{m}_n[p] \geq \alpha {}^\bullet t_{n+1}[p]$ for all $p \in P$. Since $b \in \mathbb{N}$, it is the case that $b \cdot \boldsymbol{m}_n[p] \geq b \cdot \alpha {}^\bullet t_{n+1}[p]$, and hence we are done. ☐

Proposition 1. *Let $\mathcal{N} = (P, T, F)$ be a workflow net, and let $\mathbb{D} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}_{\geq 0}\}$. Let $\mathcal{N}' = (P', T', F')$ be a workflow net obtained by applying a reduction rule R_i to $\mathcal{N}$, where $P = P' \cup P''$. The following holds.*

- Rule R_1. *We have $P'' = \{p\}$. There exists a nonempty set $R' \subseteq P'$ such that if $\{\mathsf{i} \colon 1\} \rightarrow^*_{\mathbb{D}} \boldsymbol{m}$ in $\mathcal{N}$, then $\boldsymbol{m}[p] = \sum_{r \in R'} \boldsymbol{m}[r']$. Moreover, $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$ in $\mathcal{N}$ iff $\pi(\boldsymbol{m}) \rightarrow^*_{\mathbb{D}} \pi(\boldsymbol{n})$ in $\mathcal{N}'$.*
- Rules R_2 and R_3. *We have $P'' = \emptyset$ and $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$ in $\mathcal{N}$ iff $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$ in $\mathcal{N}'$.*

– Rules R_4 and R_5. *We have* $P'' = \{p\}$. *For all* $\boldsymbol{m'}$ *and* $\boldsymbol{n'}$, $\boldsymbol{m'} \rightarrow^*_{\mathbb{D}} \boldsymbol{n'}$ *in* $\mathcal{N}'$ *iff* $\pi_0(\boldsymbol{m'}) \rightarrow^*_{\mathbb{D}} \pi_0(\boldsymbol{n'})$ *in* $\mathcal{N}$. *Further, for all* $t \in T$ *and* $p' \in P'$*: either* ${}^\bullet t[p] = 1$ *implies* ${}^\bullet t[p'] = 0$*; or* $t^\bullet[p] = 1$ *implies* $t^\bullet[p'] = 0$. *Also, for* $\mathbb{D} \neq \mathbb{Z}$, *if* $\exists \boldsymbol{m} : \{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{D}} \boldsymbol{m} \not\rightarrow^*_{\mathbb{D}} \{\mathsf{f}\colon 1\}$ *holds in* $\mathcal{N}$, *then* $\exists \boldsymbol{m'} : \{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{D}} \boldsymbol{m'} \not\rightarrow^*_{\mathbb{D}} \{\mathsf{f}\colon 1\}$ *holds in* $\mathcal{N}'$.
– Rule R_6. *We have* $P'' = \{p_2, \ldots, p_k\}$. *There exists* $p_1 \in P'$ *such that for all* $\boldsymbol{n} \in P^{\mathbb{D}}$, *if* $\sum_{i=1}^k \boldsymbol{m}[p_i] = \sum_{i=1}^k \boldsymbol{n}[p_i]$ *and* $\boldsymbol{n}[p'] = \boldsymbol{m}[p']$ *for* $p' \in P' \setminus \{p_1\}$, *then* $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$. *Moreover, if* $\boldsymbol{m}[p_i] = \boldsymbol{n}[p_i] = 0$ *for* $i > 1$, *then* $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$ *in* $\mathcal{N}$ *iff* $\pi(\boldsymbol{m}) \rightarrow^*_{\mathbb{D}} \pi(\boldsymbol{n})$ *in* $\mathcal{N}'$.

Proof. We will informally present the rules by the properties they preserve. For a formal definition of the rules, we refer to [10, Sect. 4.2]. Most arguments apply to all $\mathbb{D} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}_{\geq 0}\}$ in the same way, thus usually we will not make a cumbersome case analysis.

Rule R_1 *(place removal).* This rule removes a place $p \in P$. Thus, $P' = P \setminus \{p\}$. It is guaranteed that there exist places $\{g_1, \ldots, g_n\} \subseteq P'$ such that the number of tokens in p is the sum of tokens in those places. Hence, it suffices to define $R' \coloneqq \{g_1, \ldots, g_n\}$.

Rules R_2 *(transition removal) and* R_3 *(loop removal).* For these rules, no place is removed and the reachability relation is preserved.

Rules R_4 *(transition-place removal) and* R_5 *(place-transition removal).* These rules remove a place p and its only input (for R_4) or output (for R_5) transition t. Transition t is merged with the output (for R_4) or input (for R_5) transitions. Thus, intuitively, the new transitions in $\mathcal{N}'$ immediately consume a token whenever it was put in p. This clearly proves that $\boldsymbol{m'} \rightarrow^*_{\mathbb{D}} \boldsymbol{n'}$ in $\mathcal{N}'$ iff $\pi_0(\boldsymbol{m'}) \rightarrow^*_{\mathbb{D}} \pi_0(\boldsymbol{n'})$ in $\mathcal{N}$. Moreover, the requirements on when the rule can be applied imply either ${}^\bullet t[p] = 1 \implies {}^\bullet t[p'] = 0$; or $t^\bullet[p] = 1 \implies t^\bullet[p'] = 0$.

It remains to prove the final part when $\mathbb{D} \neq \mathbb{Z}$. Suppose there exists $\boldsymbol{m}$ such that $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{D}} \boldsymbol{m} \not\rightarrow^*_{\mathbb{D}} \{\mathsf{f}\colon 1\}$ in $\mathcal{N}$. Suppose first, that there exists $\boldsymbol{n}$ such that $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$ and $\boldsymbol{n}[p] = 0$. By the previous case, we have $\pi(\boldsymbol{n}) \not\rightarrow^*_{\mathbb{D}} \{\mathsf{f}\colon 1\}$, as otherwise we reach the contradiction $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n} \rightarrow^*_{\mathbb{D}} \{\mathsf{f}\colon 1\}$. We define $\boldsymbol{m'} \coloneqq \pi(\boldsymbol{n})$. In the second case, we can assume that for all $\boldsymbol{n}$, $\boldsymbol{m} \rightarrow^*_{\mathbb{D}} \boldsymbol{n}$ implies $\boldsymbol{n}[p] > 0$ (here we use $\mathbb{D} \neq \mathbb{Z}$). In particular, $\boldsymbol{m}[p] > 0$. Let $T_p \coloneqq \{t \in T \mid t^\bullet[p] = 1\}$. We conclude from the additional constraints on R_4 and R_5 (see [10]). These imply that in our case for every $t \in T_p$ and for all $r \in P$:

1. if $r \neq p$, then $t^\bullet[r] = 0$;
2. if ${}^\bullet t[r] = 1$ and $t' \neq t$, then ${}^\bullet t'[r] = 0$.

Let ρ be the run witnessing $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{D}} \boldsymbol{m}$. Let ρ' be the subrun of transitions in T_p that occur in ρ (it is nonempty since $\boldsymbol{m}[p] > 0$). By Item 1 we can remove (or downscale if $\mathbb{D} = \mathbb{Q}_{\geq 0}$) a suffix of $\boldsymbol{m}[p]$ transitions in ρ' (because it removes tokens only from p). We obtain a marking $\boldsymbol{m}_1$ such that: $\boldsymbol{m}_1[p] = 0$; the tokens in $\boldsymbol{m}_1[r]$ for all removed $t \in T_p$ such that ${}^\bullet t[r] = 1$ have increased accordingly; and $\boldsymbol{m}_1[p'] = \boldsymbol{m}[p]$ otherwise. We claim that $\boldsymbol{m'} = \pi(\boldsymbol{m}_1)$ satisfies the proposition.

Indeed, if there is a run $\pi(\boldsymbol{m}_1) \to_{\mathbb{D}}^* \{\mathsf{f}\colon 1\}$ in $\mathcal{N}'$ then by Item 2 we can extract from this a run $\boldsymbol{m} \to_{\mathbb{D}}^* \{\mathsf{f}\colon 1\}$ in $\mathcal{N}$, which would be a contradiction.

R_6 *(ring removal).* This rule merges a set of places $\{p_1, \ldots, p_k\} \subseteq P$ into a single place p_1[8]. Thus, $P' = P \setminus \{p_2, \ldots, p_k\}$. The conditions are that the tokens can be transferred arbitrarily between the places $p_1, \ldots, p_k$, which is enough to prove the proposition. □

A.2 Missing proofs of Section 4

Proposition 3. *The reduction rules from [10] preserve integer unboundedness.*

Proof. We will need to invoke Proposition 4 which is stated after Proposition 3 in the main text. Note that this ordering is simply for the sake of presentation, there is no circular dependency, the proof of Proposition 4 is self-contained.

By Proposition 4, being integer unbounded is equivalent to the existence of $\boldsymbol{v} > \boldsymbol{0}$ such that $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v}$. Let $\mathcal{N}$ and $\mathcal{N}'$ be the workflow nets before and after the reduction. We invoke Proposition 1 depending on the applied reduction rule, and show that $\mathcal{N}$ is integer unbounded iff $\mathcal{N}'$ is integer unbounded.

- *Rule* R_1. Suppose $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v} > \boldsymbol{0}$ in $\mathcal{N}$. We have $\pi(\boldsymbol{v}) > \boldsymbol{0}$, since $\boldsymbol{v}[p] = \sum_{r \in R'} \boldsymbol{v}[r]$. Thus, $\pi(\boldsymbol{v})[r] > \boldsymbol{0}$ for at least one $r \in R'$. The converse implication is trivial.
- *Rules* R_2 *and* R_3. This is trivial because $\to_{\mathbb{Z}}^*$ is preserved.
- *Rules* R_4 *and* R_5. We have $\boldsymbol{m'} \to_{\mathbb{Z}}^* \boldsymbol{n'}$ in $\mathcal{N}'$ iff $\pi_0(\boldsymbol{m'}) \to_{\mathbb{Z}}^* \pi_0(\boldsymbol{n'})$ in $\mathcal{N}$. Thus, if $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v'} > \boldsymbol{0}$ in $\mathcal{N}'$, then $\boldsymbol{0} \to_{\mathbb{Z}}^* \pi_0(\boldsymbol{v'}) > \boldsymbol{0}$ in $\mathcal{N}$. Conversely, suppose that $\boldsymbol{0} \to_{\mathbb{Z}}^{\rho} \boldsymbol{v} > \boldsymbol{0}$ in $\mathcal{N}$. If $\boldsymbol{v}[p] = 0$, then we are done. Otherwise, by Proposition 1 for all $t \in T$ and $p' \in P'$: either ${}^\bullet t[p] = 1 \implies {}^\bullet t[p'] = 0$; or $t^\bullet[p] = 1 \implies t^\bullet[p'] = 0$. Let us assume the former and let $T_p \coloneqq \{t \in T \mid {}^\bullet t[p] = 1\}$. By removing $\boldsymbol{v}[p]$ transitions from T_p in ρ, we get $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v'} > \boldsymbol{0}$ and $\boldsymbol{v'}[p] = 0$. Thus, $\boldsymbol{0} \to_{\mathbb{Z}}^* \pi(\boldsymbol{v'}) > \boldsymbol{0}$ in $\mathcal{N}'$ as required. In the latter case, we proceed similarly, but one need to add some transitions to ρ that will move the tokens from p to other places.
- *Rule* R_6. In this case, if $\boldsymbol{m}[p_i] = \boldsymbol{n}[p_i] = 0$ for $i > 1$, then $\boldsymbol{m} \to_{\mathbb{Z}}^* \boldsymbol{n}$ in $\mathcal{N}$ iff $\pi(\boldsymbol{m}) \to_{\mathbb{Z}}^* \pi(\boldsymbol{n})$ in $\mathcal{N}'$. Thus, $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v'} > \boldsymbol{0}$ in $\mathcal{N}'$ clearly implies $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v} > \boldsymbol{0}$ in $\mathcal{N}$. Conversely, if $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{v} > \boldsymbol{0}$ in $\mathcal{N}$, then we know that $\boldsymbol{v} \to_{\mathbb{Z}}^* \boldsymbol{v}_1$ where $\boldsymbol{v}_1[p_1] = \sum_{i=1}^{k} \boldsymbol{v}_1[p_i]$ and $\boldsymbol{v}_1[p_i] = 0$ for $i > 1$. So, $\boldsymbol{0} \to_{\mathbb{Z}}^* \pi(\boldsymbol{v}_1) > \boldsymbol{0}$ in $\mathcal{N}'$. □

Proposition 5. *A marked Petri net* $(\mathcal{N}, \boldsymbol{m})$*, where* $\mathcal{N} = (P, T, F)$*, is integer unbounded iff this system has a solution:* $\exists \boldsymbol{x} \in \mathbb{Q}_{\geq 0}^T : \sum_{t \in T} \boldsymbol{x}[t] \cdot \Delta(t) > \boldsymbol{0}$*. In particular, given a workflow net* $\mathcal{N}$*, testing integer boundedness of* $(\mathcal{N}, \{\mathsf{i}\colon 1\})$ *can be done in polynomial time.*

Proof. Let $\mathcal{N} = (P, F, T)$ be a Petri net. By Proposition 4, $(\mathcal{N}, \boldsymbol{m})$ is integer bounded iff there exists $\boldsymbol{m'} > \boldsymbol{0}$ such that $\boldsymbol{0} \to_{\mathbb{Z}}^* \boldsymbol{m'}$. The latter amounts to the

[8] In [10], p_1 is also removed and a new place p is added, but this is trivially equivalent.

existence of $\pi \in T^*$ such that $\Delta(\pi) > \mathbf{0}$. So, this is equivalent to this system: $\exists \boldsymbol{x} \in \mathbb{N}^T : \sum_{t \in T} \boldsymbol{x}[t] \cdot \Delta(t) > \mathbf{0}$. It is readily seen that this system is equivalent to the one where $\boldsymbol{x} \in \mathbb{Q}_{\geq 0}^T$. Indeed, by homogeneity ($\mathbf{0}$ on the right-hand side), a rational solution can be scaled so that it becomes an integral solution.

The polynomial time decidability of integer boundedness follows immediately from the fact that linear programming can be solved in polynomial time (*e.g.*, see [31]). ◻

Proposition 6. *The reduction rules from [10] preserve continuous soundness.*

Proof. Let $\mathcal{N}$ and $\mathcal{N}'$ be the workflow nets before and after the reduction. We invoke Proposition 1 depending on the applied reduction rule and show that $\mathcal{N}$ is continuous sound iff $\mathcal{N}'$ is continuous sound.

- *Rule R_1.* Suppose $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}' \not\rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$ in $\mathcal{N}'$. Let $\boldsymbol{m}$ be such that $\pi(\boldsymbol{m}) = \boldsymbol{m}'$ and $\boldsymbol{m}[p] = \sum_{r \in R'} \boldsymbol{m}[r']$. Then $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}$ and $\boldsymbol{m} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$ would imply $\boldsymbol{m}' \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$, which is a contradiction. The converse implication is trivial.
- *Rules R_2 and R_3.* This is trivial because $\rightarrow^*_{\mathbb{Q}_{\geq 0}}$ is preserved.
- *Rules R_4 and R_5.* Suppose $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}' \not\rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$ in $\mathcal{N}'$. We have $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \pi_0(\boldsymbol{m}')$. If $\pi_0(\boldsymbol{m}') \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$ in $\mathcal{N}$ then, since $\{\mathsf{f}\colon 1\}[p] = 0$, we obtain $\boldsymbol{m}' \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$ in $\mathcal{N}'$, which is a contradiction. The other implication is explicitly written in Proposition 1.
- *Rule R_6.* Suppose $\{\mathsf{i}\colon 1\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{m} \not\rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$ in $\mathcal{N}$. We have $\boldsymbol{m} \rightarrow^*_{\mathbb{Z}} \boldsymbol{m}_1$ where $\boldsymbol{m}_1[p_1] = \sum_{i=1}^{k} \boldsymbol{v}_1[p_i]$ and $\boldsymbol{m}_1[p_i] = 0$ for $i > 1$. If $\pi(\boldsymbol{m}_1) \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$, then $\boldsymbol{m}_1 \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$, which is a contradiction. The other implication is trivial. ◻

Theorem 2. *Continuous soundness is coNP-complete. Moreover, coNP-hardness holds even if the underlying graph of the given workflow net is acyclic.*

Proof (of coNP-hardness). Recall that, in the main text, we have defined a workflow net $\mathcal{N}_\varphi$ from a formula in DNF, and claimed that $\mathcal{N}_\varphi$ is continuously sound iff φ is a tautology. It remains to show the implication from right to left.

$\Rightarrow$) Suppose φ is a tautology. Let us first make an observation. Consider some sequence $b_1, \ldots, b_m \in \{0, 1\}$, and the marking $\boldsymbol{m} = \{p_{i,b_i}\colon 1 \mid i \in [1..m]\}$. Since φ is a tautology, there exists a clause C_j that satisfies the assignment $x_i \coloneqq b_i$. Let $i_1, \ldots, i_\ell$ be the indices of variables not occurring in C_j. It is easy to see that

$$\{\mathsf{i}\colon i\} \rightarrow^{t_{\text{init}} v_{1,b_1} \cdots v_{m,b_m}} \boldsymbol{m} \rightarrow^{c_j \overline{v}_{i_1,b_{i_1}} \cdots \overline{v}_{i_\ell,b_{i_\ell}}} \{r_i\colon 1 \mid i \in [1..m]\} \rightarrow^{t_{\text{fin}}} \{\mathsf{f}\colon 1\}.$$

By [19, Lemma 12(1)], we rescale the continuous run, *i.e.* for all $\alpha \in (0, 1]$:

$$\{\mathsf{i}\colon \alpha\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \alpha\boldsymbol{m} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \sum_{i=1}^{m} \{r_i\colon \alpha\} \rightarrow^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon \alpha\}. \tag{1}$$

Let us establish some invariants. Let $A_i \coloneqq \{\mathsf{i}, p_{i,?}, p_{i,1}, p_{i,0}, r_i, \mathsf{f}\}$ and $B_i \coloneqq \{\mathsf{i}, p_{\text{cl}}, q_i, r_i, \mathsf{f}\}$. First, for all transition $t \in T$ and all index $i \in [1..m]$, we have

$$\sum_{p \in A_i} {}^\bullet t[p] = \sum_{A_i} t^\bullet[p], \text{ and } \sum_{p \in B_i} {}^\bullet t[p] = \sum_{p \in B_i} t^\bullet[p].$$

We say that a marking $\boldsymbol{n}$ is *reachable* if $\{\mathsf{i}\colon 1\} \to^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{n}$. From the above invariants, it follows that every reachable marking $\boldsymbol{n}$ satisfies

$$\sum_{p \in A_i} \boldsymbol{n}[p] = 1 \text{ and } \sum_{p \in B_i} \boldsymbol{n}[p] = 1. \tag{2}$$

Note that, from Equation (2), every reachable marking $\boldsymbol{n}$ satisfies

$$\boldsymbol{n}[p_{i,?}] + \boldsymbol{n}[p_{i,1}] + \boldsymbol{n}[p_{i,0}] = \boldsymbol{n}[p_{\text{cl}}] + \boldsymbol{n}[q_i]. \tag{3}$$

We further have this remaining invariant for all $t \in T$ and $i, j \in [1..m]$:

$${}^\bullet t[q_i] + {}^\bullet t[r_i] + t^\bullet[q_i] + t^\bullet[r_i] = {}^\bullet t[q_j] + {}^\bullet t[r_j] + t^\bullet[q_j] + t^\bullet[r_j].$$

Since all places q_i and r_i are empty in $\{\mathsf{i}\colon 1\}$, every reachable marking $\boldsymbol{n}$ satisfies:

$$\boldsymbol{n}[q_i] + \boldsymbol{n}[r_i] = \boldsymbol{n}[q_j] + \boldsymbol{n}[r_j]. \tag{4}$$

We are ready to prove continuous soundness. Let $\boldsymbol{n}$ be a reachable marking. By Equation (1), we can assume w.l.o.g. that $\boldsymbol{n}[\mathsf{i}] = 0$, as we can move α remaining token to f. Similarly, we can assume w.l.o.g. that $\boldsymbol{n}[p_{i,?}] = 0$ for all $i \in [1..m]$ as otherwise we can fire transition $v_{i,1}$ or $v_{i,0}$ properly scaled (the choice is irrelevant). Consequently, by Equation (3), we have $\boldsymbol{n}[p_{i,1}]+\boldsymbol{n}[p_{i,0}] \geq \boldsymbol{n}[q_i]$. Therefore, by firing transitions $\overline{v}_{i,0}$ and $\overline{v}_{i,1}$, scaled appropriately, we obtain $\boldsymbol{n} \to^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{n}'$ with $\boldsymbol{n}'[q_i] = 0$ for all $i \in [1..m]$. By Equation (4), $\boldsymbol{n}'[r_i] = \boldsymbol{n}'[r_j]$ for all $i, j \in [1..m]$. Hence, by firing t_{fin} scaled by $\boldsymbol{n}'[r_1]$, we get $\boldsymbol{n}' \to^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{n}''$ where $\boldsymbol{n}''$ has zero token in each place, except possibly places $P'_{\text{var}} \coloneqq \{p_{i,b} \mid i \in [1..m], b \in \{0,1\}\}$, place p_{cl} and place f.

Let us explain how to empty $P'_{\text{var}} \cup \{p_{\text{cl}}\}$, if this is not already the case. For each $i \in [1..m]$, among places $p_{i,1}$ and $p_{i,0}$, we write $p_{i,\max}$ and $p_{i,\min}$ so that $\boldsymbol{n}''[p_{i,\max}] \geq \boldsymbol{n}''[p_{i,\min}]$ (if they are equal, then the choice is not important). Let $S \coloneqq \{p_{i,\min} \mid i \in [1..m], \boldsymbol{n}''[p_{i,\min}] > 0\}$. We consider two cases.

Case 1: $S = \emptyset$. By the left part of Equation (2), and by Equation (3), the following holds for all $i, j \in [1..m]$:

$$\boldsymbol{n}''[p_{i,1}] + \boldsymbol{n}''[p_{i,0}] = \boldsymbol{n}''[p_{j,1}] + \boldsymbol{n}''[p_{j,0}] = \boldsymbol{n}''[p_{\text{cl}}]. \tag{5}$$

Thus, there exist $\alpha \in (0,1]$ and $b_1, \ldots, b_m \in \{0,1\}$ such that $\boldsymbol{n}''[p_{\text{cl}}] = \boldsymbol{n}''[p_{i,b_i}] = \alpha$ and $\boldsymbol{n}''[p_{i,\neg b_i}] = 0$ for all $i \in [1..m]$. Since φ is a tautology, we can fire some transition c_j scaled by α, which empties place p_{cl}, and consequently P'_{var} as well by Equation (5).

Case 2: $S \neq \emptyset$. Let $i \in [1..m]$ be such that $\boldsymbol{n}''[p_{i,\min}] > 0$ is minimal, and let $\alpha \coloneqq \boldsymbol{n}''[p_{i,\min}]$. Let $\boldsymbol{n}''' \coloneqq \{p_{\text{cl}} \colon \alpha, p_{i,\min} \colon \alpha\} + \{p_{j,\max} \colon \alpha \mid j \neq i\}$. Note that $\boldsymbol{n}''' \leq \boldsymbol{n}''$. We can apply Equation (1) and obtain

$$\boldsymbol{n}'' = (\boldsymbol{n}'' - \boldsymbol{n}''') + \boldsymbol{n}''' \rightarrow^{*}_{\mathbb{Q}_{\geq 0}} (\boldsymbol{n}'' - \boldsymbol{n}''') + \{\mathsf{f} \colon \alpha\}.$$

Performing this operation decreases the size of S. Hence, it can be repeated at most m times until S becomes empty, which has been handled in case 1. ◻

A.3 Missing proofs of Section 5

Proposition 10. *Let $\mathcal{N}$ be a workflow net. It is the case that $k_{\mathcal{N},\mathbb{Z}} \leq k_{\mathcal{N},\mathbb{Q}_{\geq 0}} \leq k_{\mathcal{N}}$. Moreover, $k_{\mathcal{N},\mathbb{Z}}$ can be computed from an integer linear program $\mathcal{P}$; $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ can be obtained by computing $\min k \in \mathbb{N}_{\geq 1} : \varphi(k)$ where φ is a formula from the existential fragment of mixed linear arithmetic φ, i.e. $\exists\mathsf{FO}(\mathbb{Q}, \mathbb{Z}, <, +)$; and both $\mathcal{P}$ and φ are constructible in polynomial time from $\mathcal{N}$.*

Proof. Let $\mathcal{N} = (P, T, F)$ be a workflow net. Let us first establish $k_{\mathcal{N},\mathbb{Z}} \leq k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$. Let $\pi = \lambda_1 t_1 \cdots \lambda_n t_n$ be a continuous run such that $\{\mathsf{i} \colon k\} \rightarrow^{\pi}_{\mathbb{Q}_{\geq 0}} \{\mathsf{f} \colon k\}$ and $\boldsymbol{\pi} \in \mathbb{N}^T$. In particular, we have

$$\begin{aligned}
\{\mathsf{f} \colon k\} &= \{\mathsf{i} \colon k\} + \sum_{i \in [1..n]} \lambda_i \cdot \Delta(t_i) \\
&= \{\mathsf{i} \colon k\} + \sum_{t \in T} \sum_{i \in [1..n] : t_i = t} \lambda_i \cdot \Delta(t) \\
&= \{\mathsf{i} \colon k\} + \sum_{t \in T} \boldsymbol{\pi}[t] \cdot \Delta(t).
\end{aligned}$$

As $\boldsymbol{\pi} \in \mathbb{N}^T$, we obtain $\{\mathsf{i} \colon k\} \rightarrow^{\pi}_{\mathbb{Z}} \{\mathsf{f} \colon k\}$. Consequently, $k_{\mathcal{N},\mathbb{Z}} \leq k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$.

The inequality $k_{\mathcal{N},\mathbb{Q}_{\geq 0}} \leq k_{\mathcal{N}}$ follows immediately from the fact that $\{\mathsf{i} \colon k\} \rightarrow^{\pi} \{\mathsf{f} \colon k\}$ implies $\{\mathsf{i} \colon k\} \rightarrow^{\pi}_{\mathbb{Q}_{\geq 0}} \{\mathsf{f} \colon k\}$ (with all scaling factors set to 1).

It remains to argue that $k_{\mathcal{N},\mathbb{Z}}$ and $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$ can be obtained as described. By definition of integer reachability, $k_{\mathcal{N},\mathbb{Z}}$ is the value obtained from this program:

$$\min k \text{ subject to } k \in \mathbb{N}_{\geq 1}, \boldsymbol{x} \in \mathbb{N}^T \text{ and } \{\mathsf{i} \colon k\} + \sum_{t \in T} \boldsymbol{x}[t] \cdot \Delta(t) = \{\mathsf{f} \colon k\}.$$

For $k_{\mathcal{N},\mathbb{Q}_{\geq 0}}$, we use the fact that there is polynomial-time constructible formula $\psi_{\mathcal{N}}$ from existential linear arithmetic such that $\psi(\boldsymbol{m}, \boldsymbol{m}', \boldsymbol{x})$ holds iff there is a continuous run π that satisfies $\boldsymbol{m} \rightarrow^{\pi}_{\mathbb{Q}_{\geq 0}} \boldsymbol{m}'$ and $\boldsymbol{x} = \boldsymbol{\pi}$ [8]. So, it suffices to take

$$\varphi(k) \coloneqq \exists \boldsymbol{x} \in \mathbb{N}^T : \psi(\{\mathsf{i} \colon k\}, \{\mathsf{f} \colon k\}, \boldsymbol{x}).$$ ◻

A.4 Missing proofs of Section 6

Recall the following unproven lemma from the main text.

Lemma 2. *Let $\mathcal{N} = (P, T, F)$ be a free-choice Petri net, let $c \in \mathbb{N}_{\geq 1}$, and let $\boldsymbol{m} \in \mathbb{N}^P$. The following statements hold.*

1. *There exists a marking $\boldsymbol{m'}$ such that $\boldsymbol{m} \to^* \boldsymbol{m'}$ and $L(\boldsymbol{m'}) = F(\boldsymbol{m'})$.*
2. *If $L(\boldsymbol{m}) = F(\boldsymbol{m})$, then $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m}) = F(\boldsymbol{m})$.*
3. *If $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m})$, $c \cdot \boldsymbol{m} \to^* \{\mathsf{f}\colon c\}$ and $(\mathcal{N}, c \cdot \boldsymbol{m})$ is bounded, then $\boldsymbol{m} = \{\mathsf{f}\colon 1\}$.*

For the sake of readability, we prove each item of Lemma 2 as its own lemma.

Lemma 4. *Let $\mathcal{N} = (P, T, F)$ be a free-choice Petri net, and let $\boldsymbol{m} \in \mathbb{N}^P$. It is the case that $\boldsymbol{m} \to^* \boldsymbol{m'}$ for some marking $\boldsymbol{m'}$ such that $L(\boldsymbol{m'}) = F(\boldsymbol{m'})$.*

Proof. If $F(\boldsymbol{m}) = L(\boldsymbol{m})$ holds, then we are done by taking $\boldsymbol{m'} := \boldsymbol{m}$. Otherwise, let $t \in F(\boldsymbol{m}) \setminus L(\boldsymbol{m})$. Since t is not live in $(\mathcal{N}, \boldsymbol{m})$, there exists a marking $\boldsymbol{m'} \in \mathbb{N}^P$ that satisfies $\boldsymbol{m} \to^* \boldsymbol{m'}$ and $t \notin F(\boldsymbol{m'})$. Therefore, we have $F(\boldsymbol{m'}) \subseteq F(\boldsymbol{m}) \setminus \{t\}$. This means that $|F(\boldsymbol{m'})| < |F(\boldsymbol{m})$. Since $L(\boldsymbol{m'}) \subseteq F(\boldsymbol{m'})$, we can repeat this argument (up to $|T|$ times) until obtaining $L(\boldsymbol{m'}) = F(\boldsymbol{m'})$. ◻

For a run σ, let us define $\boldsymbol{\sigma}\colon T \to \mathbb{N}$, where for each $t \in T$, $\boldsymbol{\sigma}[t]$ is the number of times t occurs in σ.

Lemma 5. *Let $\mathcal{N}$ be a free-choice workflow net, let $c \in \mathbb{N}_{\geq 1}$, and let $\boldsymbol{m} \in \mathbb{N}^P$ be such that $L(\boldsymbol{m}) = F(\boldsymbol{m})$. It is the case that $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m}) = F(\boldsymbol{m})$.*

Proof. We first show that $F(c \cdot \boldsymbol{m}) = F(\boldsymbol{m})$, and then that $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m})$.

We trivially have $F(c \cdot \boldsymbol{m}) \supseteq F(\boldsymbol{m})$. For the sake of contradiction, suppose there exists a transition $t \in F(c \cdot \boldsymbol{m})$ such that $t \notin F(\boldsymbol{m})$. Let σ_1 be a run such that $c \cdot \boldsymbol{m} \to^{\sigma_1 t}$. Without loss of generality, we may assume that $\boldsymbol{\sigma_1} \subseteq F(\boldsymbol{m})$. Indeed, if there is some $t' \in \boldsymbol{\sigma_1}$ such that $t' \notin F(\boldsymbol{m})$, then we can shorten σ_1 and take the shortened run which enables t' instead.

Let $\sigma_1 = t_1 t_2 \cdots t_n$. Recall that $t_i \in L(\boldsymbol{m}) = F(\boldsymbol{m})$ for each t_i, that is, from any marking reachable from $\boldsymbol{m}$, we can reach a marking that enables t_i. Therefore, we can define a run $\sigma_2 := \phi_1 t_1 \phi_2 t_2 \cdots \phi_n t_n$, where ϕ_i is a run from $\boldsymbol{m} + \Delta(\phi_1 t_1 \cdots \phi_{i-1} t_{i-1})$ that enables t_i.

If there exists a transition s in the run σ_2 such that $\text{supp}({}^\bullet s) \cap \text{supp}({}^\bullet t) \neq \emptyset$, then ${}^\bullet s = {}^\bullet t$ as $\mathcal{N}$ is free-choice. Hence, since $s \in F(\boldsymbol{m})$, we obtain $t \in F(\boldsymbol{m})$, which is a contradiction. Thus no transition in σ_2 can consume tokens from places in $\text{supp}({}^\bullet t)$. Since $c \cdot \boldsymbol{m} \to^{\sigma_1 t}$, we know that

$$\text{supp}({}^\bullet t) \subseteq \text{supp}(c \cdot \boldsymbol{m}) \cup \bigcup_{i=1}^{n} \text{supp}(t_i^\bullet) = \text{supp}(\boldsymbol{m}) \cup \bigcup_{i=1}^{n} \text{supp}(t_i^\bullet).$$

Altogether, this means that the transitions t_i put enough tokens such that all places in $\text{supp}({}^\bullet t)$ are marked, and that σ_2 cannot consume any of these tokens. Therefore, $\boldsymbol{m} \to^{\sigma_2 t}$, which is a contradiction.

It remains to prove that $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m})$. We have $L(\boldsymbol{m}) \subseteq L(c \cdot \boldsymbol{m}) \subseteq F(c \cdot \boldsymbol{m})$. Since $L(\boldsymbol{m}) = F(\boldsymbol{m}) = F(c \cdot \boldsymbol{m})$, these inclusions are in fact equalities, and we are done. ◻

Lemma 6. *Let $\mathcal{N} = (P, F, T)$ be a free-choice workflow net, let $c \in \mathbb{N}_{\geq 1}$ and let $\boldsymbol{m} \in \mathbb{N}^P$ be such that $L(c \cdot \boldsymbol{m}) = F(c \cdot \boldsymbol{m})$. If $c \cdot \boldsymbol{m} \to^* \{\mathsf{f}\colon c\}$ and $(\mathcal{N}, c \cdot \boldsymbol{m})$ is bounded, then $\boldsymbol{m} = \{\mathsf{f}\colon 1\}$.*

Proof. Recall that no transition of a workflow net consumes from f, *i.e.* ${}^\bullet t[\mathsf{f}] = 0$ for all $t \in T$. Thus, we either have $\boldsymbol{m}[\mathsf{f}] = 0$ or $\boldsymbol{m}[\mathsf{f}] = 1$.

If $\boldsymbol{m}[\mathsf{f}] = 0$, then there is some transition $t \in F(c \cdot \boldsymbol{m})$ such that $f \in t^\bullet[\mathsf{f}] > 0$. Since $t \in L(c \cdot \boldsymbol{m})$, it follows that from $c \cdot \boldsymbol{m}$, we can reach $\boldsymbol{m}'$ with $\boldsymbol{m}'[\mathsf{f}]$ abritrarily large, as t puts a token into f and can be fired arbitrarily often from $c \cdot \boldsymbol{m}$. This contradicts the fact that $(\mathcal{N}, c \cdot \boldsymbol{m})$ is bounded. Hence, $\boldsymbol{m}[\mathsf{f}] = 1$. We can write $\boldsymbol{m}$ as $\boldsymbol{m} = \{\mathsf{f}\colon 1\} + \boldsymbol{m}'$ where $\boldsymbol{m}'[\mathsf{f}] = 0$. We have $c \cdot \boldsymbol{m} = \{\mathsf{f}\colon c\} + c \cdot \boldsymbol{m}'$. If $\boldsymbol{m}' = \boldsymbol{0}$, then we are done. Otherwise, we obtain a contradiction. Indeed, it cannot be the case that $\{\mathsf{f}\colon c\} + c \cdot \boldsymbol{m}' \to^* \{\mathsf{f}\colon c\}$, as every transition of a workflow net produces at least one token (and none consumes from f). ◻

Lemma 3. *Let $\mathcal{N}$ be a workflow net. If $\mathcal{N}$ is continuously sound, then $(\mathcal{N}, \{\mathsf{i}\colon k\})$ is bounded for all $k \in \mathbb{N}_{\geq 1}$.*

Proof. Assume for contradiction that there exists $k \in \mathbb{N}_{\geq 1}$ such that $(\mathcal{N}, \{\mathsf{i}\colon k\})$ is unbounded, but $\mathcal{N}$ is continuously sound. There exist marking $\boldsymbol{m}$ and $\boldsymbol{m}' > \boldsymbol{m}$ such that $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m} \to^* \boldsymbol{m}'$. By Lemma 1, we have $\{\mathsf{i}\colon 1\} \to^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{n} \to^*_{\mathbb{Q}_{\geq 0}} \boldsymbol{n}'$, where $\boldsymbol{n} \coloneqq (1/k) \cdot \boldsymbol{m}$ and $\boldsymbol{n}' \coloneqq (1/k) \cdot \boldsymbol{m}'$. As $\mathcal{N}$ is continuously sound, it must hold that $\boldsymbol{n} \to^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\}$. It follows that

$$\boldsymbol{n}' = \boldsymbol{n} + (\boldsymbol{n}' - \boldsymbol{n}) \to^*_{\mathbb{Q}_{\geq 0}} \{\mathsf{f}\colon 1\} + (\boldsymbol{n}' - \boldsymbol{n}).$$

This contradicts the assumption that $\mathcal{N}$ is continuously sound, as each transition of a workflow net produces at least one token, and none consumes from f. ◻

A.5 Missing definition of the arc weight encoding of Section 7

Recall that under our definition, Petri nets do not have arc weights as $F\colon ((P \times T) \cup (T \times P)) \to \{0, 1\}$. Petri nets with arc weights are defined exactly as Petri nets but with $F\colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$. An example of the arc weight encoding described in the main text is shown in Figure 7.

In this section, we will use t^{-1} to denote the reverse transition of transition t, as done in the coNP membership proof of Theorem 2.

Formally, to simulate a transition t, we add places $P_{p,t}$ and transitions $T_{p,t}$ for each place p with $b \coloneqq {}^\bullet t[p] > 1$, and places $P'_{p,t}$ and transitions $T'_{p,t}$ for each place p with $b' \coloneqq t^\bullet[p] > 1$.

From now on, when we define a transition t, we assume that ${}^\bullet t[p'] = 0$ and $t^\bullet[p'] = 0$ for each place p' except those given explicitly. We define $P_{p,t}$ as follows. We denote by $b_1, b_2, \ldots, b_n$ the binary representation of b, that is, $b = \sum_{i=1}^{n} b_i \cdot 2^{i-1}$, and similarly $b'_1, b'_2, \ldots, b'_{n'}$ for b'. The set $P_{p,t}$ consists of $2n - 1$ places. For every $i \in [1..n-1]$, we add two places l_i and r_i; and an additional place r_n. The set $T_{p,t}$ contains the following transitions:

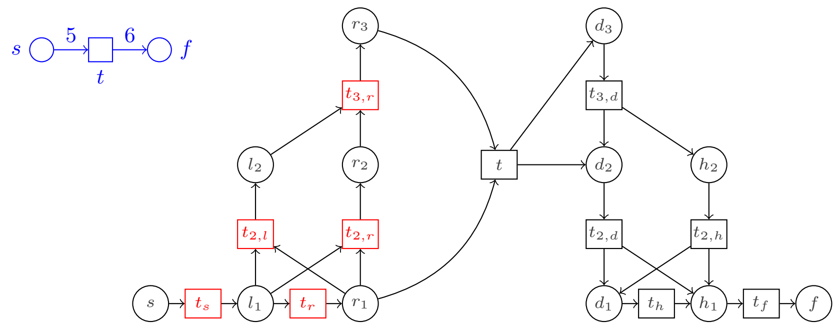

Fig. 7. *Top left (in blue):* A Petri net $\mathcal{N}$ with arc weights. *Center:* A Petri net $\mathcal{N}_{\text{enc}}$ without arc weights that simulates behaviour of $\mathcal{N}$. For each transition colored in red, the reverse transition is also part of $\mathcal{N}_{\text{enc}}$, and is merely not drawn to avoid overcrowding the figure. For ease of presentation, places and transitions of $\mathcal{N}_{\text{enc}}$ contain their names (not values).

- t_p, where ${}^\bullet t_p[p] = t_p^\bullet[l_1] \coloneqq 1$;
- t_r, as well as its reverse t_r^{-1}, where ${}^\bullet t_r[l_1] = t_r^\bullet[r_1] \coloneqq 1$;
- for each $i \in [2..n-1]$ the transitions $t_{i,l}$, $t_{i,r}$ and their reverses $t_{i,l}^{-1}$, $t_{i,r}^{-1}$, where ${}^\bullet t_{i,l}[r_{i-1}] = {}^\bullet t_{i,l}[l_{i-1}] \coloneqq 1$, ${}^\bullet t_{i,r} = {}^\bullet t_{i,l}$, and $t_{i,l}^\bullet[l_i] = t_{i,r}^\bullet[r_i] \coloneqq 1$,
- the transition $t_{n,r}$ and its reverse $t_{n,r}^{-1}$, where ${}^\bullet t_{n,r}[l_{n-1}] = {}^\bullet t_{n,r}[r_{n-1}] \coloneqq 1$ and $t_{n,r}^\bullet[r_n] \coloneqq 1$.

We further redefine t to have ${}^\bullet t[p] \coloneqq 0$ and ${}^\bullet t[r_i] \coloneqq 1$ for all i such that $b_i = 1$.

The set $P'_{p,t}$ consists of $2n'-1$ places. We have d_i and h_i for each $i \in [1..n'-1]$, and an additional place d'_n. The set $T'_{p,t}$ contains the following transitions:

- t_p, where ${}^\bullet t_p[h_1] = t_p^\bullet[p] \coloneqq 1$,
- $t_{1,h}$, where ${}^\bullet t_{1,h}[d_1] = t_{1,h}^\bullet[h_1] \coloneqq 1$,
- for each $i \in [2..n-1]$, the transitions $t_{i,d}$ and $t_{i,h}$, where ${}^\bullet t_{i,d}[d_i] = {}^\bullet t_{i,h}[h_i] \coloneqq 1$, $t_{i,d}^\bullet[d_{i-1}] = t_{i,d}^\bullet[h_{i-1}] \coloneqq 1$, and $t_{i,h}^\bullet \coloneqq t_{i,d}^\bullet$.

We further redefine t to have $t^\bullet[p] \coloneqq 0$ and $t^\bullet[d_i] \coloneqq 1$ for each i such that $b'_i = 1$.

Given a Petri net $\mathcal{N} = (P, T, F)$, let us denote by $\mathcal{N}_{\text{enc}} = (P', T', F')$ the transformed $\mathcal{N}$ where all transitions with arc weights are modified by the gadget defined above. To avoid any confusion, we denote markings in $\mathcal{N}$ as $\boldsymbol{m}$ and $\boldsymbol{m'}$, and markings in $\mathcal{N}_{\text{enc}}$ as $\boldsymbol{n}$ and $\boldsymbol{n'}$. As $\mathcal{N}_{\text{enc}}$ does not remove (but only adds) places, we may treat markings on $\mathcal{N}$ as markings on $\mathcal{N}_{\text{enc}}$, where all places in $P' \setminus P$ are marked with zero token.

Recall that $\boldsymbol{\sigma}$ is a vector mapping each transition t to the number of times t is used in run σ. In the following, let $p \in P$ and $t \in T$ be such that ${}^\bullet t[p] = b \geq 1$. Let $b_1, \ldots, b_n$ be the binary representation of b. Furthermore, let $P_{p,t}$ and $T_{p,t}$ be defined as above.

We are ready to state some helpful lemmas.

Lemma 7. *Let $i \in [1..n]$. We have $\{p\colon 2^{i-1}\} \to^\sigma \{r_i\colon 1\}$ in $\mathcal{N}_{\text{enc}}$ with $\text{supp}(\boldsymbol{\sigma}) \subseteq T_{p,t}$. Further, if $i < n$, then $\{p\colon 2^{i-1}\} \to^{\sigma'} \{l_i\colon 1\}$ in $\mathcal{N}_{\text{enc}}$ with $\text{supp}(\boldsymbol{\sigma'}) \subseteq T_{p,t}$.*

Proof. We proceed by induction. For $i = 1$, we have

$$\{p\colon 2^{1-1}\} = \{p\colon 1\} \rightarrow^{t_p} \{l_1\colon 1\} \rightarrow^{t_r} \{r_1\colon 1\}.$$

For $i > 1$, we have $\{p\colon 2^{i-1}\} = \{p\colon 2^{i-2} + 2^{i-2}\} \rightarrow^* \{r_{i-1}\colon 1, l_{i-1}\colon 1\}$ by the induction hypothesis. Thus, we have $\{r_{i-1}\colon 1, l_{i-1}\colon 1\} \rightarrow^{t_{i,r}} \{r_i\colon 1\}$. If $i < n$, then we additionally have $\{r_{i-1}\colon 1, l_{i-1}\colon 1\} \rightarrow^{t_{i,l}} \{l_i\colon 1\}$. We conclude the proof by pointing out that for all i, $t_p, t_r, t_{i,r}, t_{i,l} \in T_{p,t}$. □

The proof of the lemma below follows by the fact that all transitions of $T_{p,t}$ are reversible.

Lemma 8. *Let $i \in [1..n]$. We have $\{r_i\colon 1\} \rightarrow^{\sigma} \{p\colon 2^{i-1}\}$ in $\mathcal{N}_{\text{enc}}$ with $\text{supp}(\boldsymbol{\sigma}) \subseteq T_{p,t}$. Further, if $i < n$, then $\{l_i\colon 1\} \rightarrow^{\sigma'} \{p\colon 2^{i-1}\}$ in $\mathcal{N}_{\text{enc}}$ with $\text{supp}(\boldsymbol{\sigma'}) \subseteq T_{p,t}$.*

For the next lemma, let $p \in P$ and $t \in T$ be such that $t^\bullet[p] = b \geq 1$. Let $b_1, \ldots, b_n$ be the binary representation of b. Let $P'_{p,t}$ and $T'_{p,t}$ be as defined above.

Lemma 9. *Let $i \in [1..m]$. We have $\{d_i\colon 1\} \rightarrow^{\sigma} \{p\colon 2^{i-1}\}$ in $\mathcal{N}_{\text{enc}}$ with $\text{supp}(\boldsymbol{\sigma}) \subseteq T'_{p,t}$. Further, if $i < n$, then $\{h_i\colon 1\} \rightarrow^{\sigma'} \{p\colon 2^{i-1}\}$ in $\mathcal{N}_{\text{enc}}$ with $\text{supp}(\boldsymbol{\sigma'}) \subseteq T'_{p,t}$.*

Proof. We proceed by induction on i. If $i = 1$, then we have $2^{i-1} = 1$ and hence $\{d_1\colon 1\} \rightarrow^{t_h} \{h_1\colon 1\} \rightarrow^{t_p} \{p\colon 1\}$.

For $i > 1$, we have $\{d_i\colon 1\} \rightarrow^{t_{i,d}} \{d_{i-1}\colon 1, h_{i-1}\colon 1\}$. If $i < n$, then we additionally have $\{h_i\colon 1\} \rightarrow^{t_{i,h}} \{d_{i-1}\colon 1, h_{i-1}\colon 1\}$. It follows from the induction hypothesis that $\{d_{i-1}\colon 1, h_{i-1}\colon 1\} \rightarrow^{\sigma\sigma'} \{p\colon 2^{i-2} + 2^{i-2}\} = \{p\colon 2^{i-1}\}$. We conclude by pointing out that, for all i, we have $t_p, t_h, t_{i,d}, t_{i,h} \in T'_{p,t}$. □

Definition 1. *Let $U \subseteq T$. A vector $\boldsymbol{x}\colon P \to \mathbb{Q}$ is a* place invariant *over U if the following holds for all $t \in U$:*

$$\sum_{p \in P} {}^\bullet t[p] \cdot \boldsymbol{x}[p] = \sum_{p \in P} t^\bullet[p] \cdot \boldsymbol{x}[p]. \tag{6}$$

Proposition 11 (adapted from [15, Prop. 2.27]). *Let $U \subseteq T$ and let $\boldsymbol{x}$ be a place invariant over U. If $\boldsymbol{m} \rightarrow^{\sigma} \boldsymbol{m}'$ with $\text{supp}(\boldsymbol{\sigma}) \subseteq U$, then $\boldsymbol{x} \cdot \boldsymbol{m} = \boldsymbol{x} \cdot \boldsymbol{m}'$.*

Let us define the vector $I_{p,t}$ with $I_{p,t}[p] \coloneqq 1$, $I_{p,t}[r_i] \coloneqq 2^{i-1}$ and $I_{p,t}[l_i] \coloneqq 2^{i-1}$, where r_i and l_i are the places previously defined in $P_{p,t}$. It is easy to see that $I_{p,t}$ is a place invariant of $T_{p,t}$.

Let $R \coloneqq \{t \in T \mid {}^\bullet t[p] \geq 2\}$ and $S \coloneqq \{t \in T \mid t^\bullet[p] \geq 2\}$. We further define the vector $I_p\colon \{p\} \cup \bigcup_{t \in R} P_{p,t} \cup \bigcup_{t \in S} P'_{p,t} \to \mathbb{Q}$, where $P_{p,t} = \emptyset$ if ${}^\bullet t[p] \leq 1$ and $P'_{p,t} = \emptyset$ if $t^\bullet[p] \leq 1$. We define $I_p[p] \coloneqq 1$ and $I_p[p'] \coloneqq 2^{i-1}$ if $p' \in \{r_i, l_i, d_i, h_i\}$ for some i. Note that this is well-defined by our choice of domain of I_p. It is easy to convince oneself that I_p is a place invariant of $T' \setminus T$.

We introduce some notation. For a transition $t \in T$, let $G \coloneqq \{p \in P \mid {}^\bullet t[p] \geq 2\}$ and $H \coloneqq \{p \in P \mid t^\bullet[p] \geq 2\}$. For a place $p \in G$, we write $b(p) \coloneqq {}^\bullet t[p]$. For $i \in \mathbb{N}$, we write $n(i)$ to denote the number of bits in the binary representation of

i. Let $b_1(p), \ldots, b_{n(b(p))}(p)$ denote the bits of the binary representation of $b(p)$. Let $r_i(p)$ denote the place r_i in $P_{p,t}$. Similarly, given $p \in H$, we write $c(p) \coloneqq t^\bullet[p]$, we let $c_1(p), \ldots, c_{n(c(p))}(p)$ be the bits of the binary representation of $c(p)$, and we further write $d_i(p)$ to denote the place d_i of $P'_{p,t}$. In the following, we denote by t the transition in $\mathcal{N}$, and by t' the corresponding transition in $\mathcal{N}_{\text{enc}}$.

Lemma 10. *Let $t \in T$ and let $\boldsymbol{m}, \boldsymbol{m'}$ be markings of $\mathcal{N}$ with $\boldsymbol{m'} = \boldsymbol{m} + \Delta(t)$. It holds that $\boldsymbol{m} \rightarrow^t \boldsymbol{m'}$ in $\mathcal{N}$ iff $\boldsymbol{m} \rightarrow^{\pi t' \pi'} \boldsymbol{m'}$ in $\mathcal{N}_{\text{enc}}$, where* $\text{supp}(\boldsymbol{\pi}) \subseteq \bigcup_{p \in G} T_{p,t}$ *and* $\text{supp}(\boldsymbol{\pi'}) \subseteq \bigcup_{p \in H} T'_{p,t}$.

Proof. $\Rightarrow$) By definition of $\mathcal{N}_{\text{enc}}$, $\boldsymbol{m}[p] \geq {}^\bullet t'[p]$ for all $p \in P \setminus G$. By definition of $\mathcal{N}_{\text{enc}}$, it holds that ${}^\bullet t'[r_i(p)] = b_i(p)$ for all $i \in [1..n(b(p))]$. Note that $\boldsymbol{m}[p] \geq b(p) = \sum_{i=1}^{n(b(p))} 2^{i-1} \cdot b_i(p)$. Thus, it follows from Lemma 7 that $\{p \colon b(p)\} \rightarrow^\sigma \sum_{i=1}^{n(b(p))} \{r_i(p) \colon b_i(p)\}$. So, in particular,

$$\boldsymbol{m}[p] \rightarrow^\sigma \boldsymbol{m} - \{p \colon b(p)\} + \sum_{i=1}^{n(b(p))} \{r_i(p) \colon b_i(p)\}.$$

Since the transitions in σ do not have an effect on places other than $P_{p,t} \cup \{p\}$, we can invoke Lemma 7 individually for each $p \in G$, and thus obtain

$$\boldsymbol{m} \rightarrow^{\sigma_1 \cdots \sigma_{|G|}} \boldsymbol{m} + \sum_{p \in G} \sum_{i=1}^{n(b(p))} \{r_i(p) \colon b_i(p)\} - \{p \colon b(p)\},$$

where $\text{supp}(\boldsymbol{\sigma_1}), \ldots, \text{supp}(\boldsymbol{\sigma_{|G|}}) \subseteq \bigcup_{p \in G} T_{p,t}$. By definition, t' is enabled in this marking and its firing leads to

$$\begin{aligned} \boldsymbol{m} - \sum_{p \in G} \{p \colon b(p)\} - \sum_{p \in P \setminus G} \{p \colon {}^\bullet t[p]\} + \sum_{p \in P \setminus H} \{p \colon t^\bullet[p]\} + \sum_{p \in H} \sum_{i=1}^{n(c(p))} \{d_i(p) \colon c_i(p)\} \\ = \boldsymbol{m} - {}^\bullet t + \sum_{p \in P \setminus H} \{p \colon t^\bullet[p]\} + \sum_{p \in H} \sum_{i=1}^{n(c(p))} \{d_i(p) \colon c_i(p)\}. \end{aligned}$$

Let us denote the latter marking as $\boldsymbol{m'}$. By invoking Lemma 9 individually on each $d_i(p)$, it follows that for each $p \in H$:

$$\begin{aligned} \boldsymbol{m'} \rightarrow^{\sigma'_1 \cdots \sigma'_{|H|}} \boldsymbol{m} - {}^\bullet t + \sum_{p \in P \setminus H} \{p \colon t^\bullet[p]\} + \sum_{p \in H} \sum_{i=1}^{n(c(p))} \{p \colon 2^{i-1} c_i(p)\} = \\ \boldsymbol{m} - {}^\bullet t + \sum_{p \in P \setminus H} \{p \colon t^\bullet[p]\} + \sum_{p \in H} \{p \colon t^\bullet[p]\} = \\ \boldsymbol{m} - {}^\bullet t + t^\bullet = \boldsymbol{m} + \Delta(t). \end{aligned}$$

We conclude this direction by noting that $\text{supp}(\boldsymbol{\sigma_1}), \ldots, \text{supp}(\boldsymbol{\sigma_{|H|}}) \subseteq \bigcup_{p \in H} T'_{p,t}$ by Lemma 9.

$\Leftarrow$) We have $\boldsymbol{m} \rightarrow^{\sigma t' \sigma'} \boldsymbol{m}'$. Let us denote by $\boldsymbol{m}_1$ the marking such that $\boldsymbol{m} \rightarrow^{\sigma} \boldsymbol{m}_1$. It must be the case that

$$\boldsymbol{m}_1 \geq {}^\bullet t' = \sum_{p \in P \setminus G} {}^\bullet t[m] + \sum_{p \in G} \sum_{i \in n(b(p))} \{r_i(p) \colon b_i(p)\}.$$

Recall that for each $p \in G$, $I_{p,t}$ is a place invariant of $T_{p,t}$. In particular, among transitions from $T' \setminus T$, places in $\{p\} \cup P_{p,t}$ are only affected by transitions in $T_{p,t}$. So, $I_{p,t} \cdot \boldsymbol{m} = I_{p,t} \cdot \boldsymbol{m}_1$ by Proposition 11. Since $\boldsymbol{m}_1 \geq \sum_{i \in n(b(p))} \{r_i(p) \colon b_i(p)\}$, we have $I_{p,t} \cdot \boldsymbol{m}_1 \geq \sum_{i \in n(b(p))} 2^{i-1} b_i(p)$. Thus, the same must hold for $I_{p,t} \cdot \boldsymbol{m}$. But among places in $\{p\} \cup P_{p,t}$, $\boldsymbol{m}$ marks only p, as it is (by projection) a marking of $\mathcal{N}$. Since $I_{p,t}[p] = 1$, it must hold that $\boldsymbol{m}[p] \geq \sum_{i \in n(b(p))} 2^{i-1} b_i(p) = b(p)$, where the last equality follows from the fact that $b_1(p), \ldots, b_{n(b(p))}(p)$ is the binary representation of $b(p)$. So, $\boldsymbol{m}[p] \geq {}^\bullet t[p]$ holds by definition of $b(p)$. Therefore, $\boldsymbol{m}$ enables t, and consequently $\boldsymbol{m} \rightarrow^t \boldsymbol{m} + \Delta(t) = \boldsymbol{m}'$, and we are done. ☐

Lemma 11. *Let $\boldsymbol{m}, \boldsymbol{m}'$ be markings of $\mathcal{N}$. If $\boldsymbol{m} \rightarrow^{\sigma} \boldsymbol{m}'$ in $\mathcal{N}_{\text{enc}}$ and* $\text{supp}(\boldsymbol{\sigma}) \subseteq T' \setminus T$, *then $\boldsymbol{m} = \boldsymbol{m}'$.*

Proof. We argue for each place $p \in P$ individually that $\boldsymbol{m}[p] = \boldsymbol{m}'[p]$.

Recall that I_p is a place invariant over $T' \setminus T$. Therefore, $I_p \cdot \boldsymbol{m} = I_p \cdot \boldsymbol{m}'$ by Proposition 11. Note also that in the domain of I_p, the only place in P is p. Since $\boldsymbol{m}$ and $\boldsymbol{m}'$ are markings of $\mathcal{N}$, and consequently all places in the domain of I_p other than p must be unmarked, it follows that $I_p[p] \cdot \boldsymbol{m}[p] = I_p[p] \cdot \boldsymbol{m}'[p]$. Thus, $\boldsymbol{m}[p] = \boldsymbol{m}'[p]$. ☐

Lemma 12. *Let $\boldsymbol{m}, \boldsymbol{m}'$ be markings of $\mathcal{N}$. If $\boldsymbol{m} \rightarrow^* \boldsymbol{m}'$ in $\mathcal{N}_{\text{enc}}$, then $\boldsymbol{m} \rightarrow^* \boldsymbol{m}'$ in $\mathcal{N}$.*

Proof. Let $\boldsymbol{m} \rightarrow^{\pi} \boldsymbol{m}'$. If $\text{supp}(\boldsymbol{\pi}) \subseteq T' \setminus T$, then $\boldsymbol{m} = \boldsymbol{m}'$ by Lemma 11, and we are done. So, assume that $t \in T$ for some $t \in \pi$. We factor run π so that $\pi = \sigma_1 t_1 \sigma_1' \cdots \sigma_n t_n \sigma_n'$ with $t_1, \ldots, t_n \in T$ and

$$\text{supp}(\boldsymbol{\sigma_1}), \text{supp}(\boldsymbol{\sigma_1'}), \ldots, \text{supp}(\boldsymbol{\sigma_n}), \text{supp}(\boldsymbol{\sigma_n'}) \subseteq T \setminus T'.$$

It follows from Lemma 10 that $\{\mathsf{i} \colon 1\} \rightarrow^{t_1 \cdots t_n} \{\mathsf{f} \colon k\}$. ☐

Proposition 12. *For any workflow net $\mathcal{N}$ and any $k \in \mathbb{N}_{\geq 1}$, $\mathcal{N}$ is k-quasi-sound iff $\mathcal{N}_{\text{enc}}$ is k-quasi-sound.*

Proof. This follows immediately from Lemmas 10 and 12. ☐

Proposition 13. *For any workflow net $\mathcal{N}$ and any $k \in \mathbb{N}_{\geq 1}$, $\mathcal{N}$ is k-sound iff $\mathcal{N}_{\text{enc}}$ is k-sound.*

Proof. $\Rightarrow$) Assume $\mathcal{N}$ is k-sound. Let $\boldsymbol{m}$ be a marking of $\mathcal{N}_{\text{enc}}$ such that $\{\mathsf{i} \colon k\} \rightarrow^* \boldsymbol{m}$ in $\mathcal{N}_{\text{enc}}$. If $\boldsymbol{m}$ is also a marking of $\mathcal{N}$, then $\{\mathsf{i} \colon k\} \rightarrow^* \boldsymbol{m}$ in $\mathcal{N}$ by Lemma 12. Thus, $\boldsymbol{m} \rightarrow^* \{\mathsf{f} \colon k\}$ in $\mathcal{N}$ by k-soundness, and $\boldsymbol{m} \rightarrow^* \{\mathsf{f} \colon k\}$ in $\mathcal{N}_{\text{enc}}$ by Lemma 10. If $\boldsymbol{m}$ is not a marking on $\mathcal{N}$, then, for each place $p \in P' \setminus P$, we can invoke Lemmas 8 and 9 in order to obtain a marking $\boldsymbol{m}'$ which marks only places in P. So, we have $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m} \to^* \boldsymbol{m}'$ in $\mathcal{N}_{\text{enc}}$, and it follows by Lemma 12 that $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}'$ in $\mathcal{N}$. Thus, $\boldsymbol{m}' \to^* \{\mathsf{f}\colon k\}$ in $\mathcal{N}$ by k-soundness, and $\boldsymbol{m}' \to^* \{\mathsf{f}\colon k\}$ in $\mathcal{N}_{\text{enc}}$ by Lemma 10, which shows that $\mathcal{N}_{\text{enc}}$ is k-sound.

$\Leftarrow$) Assume $\mathcal{N}_{\text{enc}}$ is k-sound. Let $\boldsymbol{m}$ be a marking of $\mathcal{N}$ such that $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}$ in $\mathcal{N}$. If follows from Lemma 10 that $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}$ in $\mathcal{N}_{\text{enc}}$. By k-soundness of $\mathcal{N}_{\text{enc}}$, we have $\boldsymbol{m} \to^* \{\mathsf{f}\colon k\}$ in $\mathcal{N}_{\text{enc}}$. Thus, $\boldsymbol{m} \to^* \{\mathsf{f}\colon k\}$ in $\mathcal{N}$ by Lemma 12. □

A.6 Missing proofs of Section 7

Let us prove the properties claimed about the instances of Figure 4.

Proposition 14. *It is the case that*

1. *$\mathcal{N}_c$ is c-unsound and k-sound for all $k \in [1..c-1]$.*
2. *$\mathcal{N}_{\textit{sound-c}}$ is kc-sound for all $k \in \mathbb{N}_{\geq 1}$,*
3. *$\mathcal{N}_{\neg\textit{quasi-c}}$ is not structurally quasi-sound, and*
4. *$\mathcal{N}_{\neg\textit{sound-c}}$ is (mc)-quasi-sound for all $m \in \mathbb{N}_{\geq 1}$, not k-quasi-sound for any other number $k \in \mathbb{N}_{\geq 1}$, and not structurally sound.*

Proof. Items 2 and 3. They follow from the definitions of the unique transition.

Item 1. We first focus on k-soundness. Let $k \in [1..c-1]$ and let $\boldsymbol{m}$ be a marking such that $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}$. We must show that $\boldsymbol{m} \to^* \{\mathsf{f}\colon k\}$.

Recall the definition of a place invariant from Definition 1.

Let $\boldsymbol{x}[\mathsf{i}] \coloneqq c+1$, $\boldsymbol{x}[p] \coloneqq 1$, $\boldsymbol{x}[r] \coloneqq c$ and $\boldsymbol{x}[\mathsf{f}] \coloneqq c+1$. It is readily seen that $\boldsymbol{x}$ is a place invariant. Recall Proposition 11: for any two markings $\boldsymbol{n}$ and $\boldsymbol{n}'$, if $\boldsymbol{n} \to^* \boldsymbol{n}'$, then $\boldsymbol{x} \cdot \boldsymbol{n} = \boldsymbol{x} \cdot \boldsymbol{n}'$. Since $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}$, we have $\boldsymbol{x} \cdot \{\mathsf{i}\colon k\} = (c+1) \cdot k = \boldsymbol{x} \cdot \boldsymbol{m}$.

From marking $\boldsymbol{m}$, transition t_i can be fired $\boldsymbol{m}[\mathsf{i}]$ times, which leads to marking

$$\boldsymbol{m}_1 \coloneqq \{p\colon \boldsymbol{m}[p] + (c+1) \cdot \boldsymbol{m}[\mathsf{i}], r\colon \boldsymbol{m}[r], \mathsf{f}\colon \boldsymbol{m}[\mathsf{f}]\}.$$

From $\boldsymbol{m}_1$, transition t_r can be fired $\boldsymbol{m}_1[\mathsf{i}] \div c$ times, which leads to marking

$$\boldsymbol{m}_2 \coloneqq \{p\colon \boldsymbol{m}_1[p] \bmod c, r\colon \boldsymbol{m}_1[r] + \boldsymbol{m}_1[p] \div c, \mathsf{f}\colon \boldsymbol{m}_1[\mathsf{f}]\}.$$

Recall that from place invariant $\boldsymbol{x}$, we have

$$(c+1) \cdot k = (c+1) \cdot \boldsymbol{m}[\mathsf{i}] + \boldsymbol{m}[p] + c \cdot \boldsymbol{m}[r] + (c+1) \cdot \boldsymbol{m}[\mathsf{f}].$$

By reorganizing this equation, we obtain

$$\boldsymbol{m}[p] + \boldsymbol{m}[i] = (c+1)(k - \boldsymbol{m}[\mathsf{f}]) - c \cdot (\boldsymbol{m}[\mathsf{i}] + \boldsymbol{m}[r]). \tag{7}$$

This means that

$$\begin{aligned}
\boldsymbol{m}_2[p] &= \boldsymbol{m}_1[p] \bmod c && \text{(by def. of } \boldsymbol{m}_2\text{)} \\
&= (\boldsymbol{m}[p] + (c+1) \cdot \boldsymbol{m}[i]) \bmod c && \text{(by def. of } \boldsymbol{m}_1\text{)} \\
&= (\boldsymbol{m}[p] + \boldsymbol{m}[i]) \bmod c \\
&= k - \boldsymbol{m}[\mathsf{f}] && \text{(by (7))}. \qquad (8)
\end{aligned}$$

Since $\{\mathsf{i}\colon k\} \to^* \boldsymbol{m}_2$, from place invariant $\boldsymbol{x}$, we obtain

$$(c+1) \cdot k = (c+1) \cdot \boldsymbol{m}_2[\mathsf{i}] + \boldsymbol{m}_2[p] + c \cdot \boldsymbol{m}_2[r] + (c+1) \cdot \boldsymbol{m}_2[\mathsf{f}].$$

By reorganizing this equation, we obtain

$$c \cdot \boldsymbol{m}_2[r] = (c+1) \cdot (k - \boldsymbol{m}_2[\mathsf{i}] - \boldsymbol{m}_2[\mathsf{f}]) - \boldsymbol{m}_2[p]. \tag{9}$$

This means that

$$\begin{aligned} c \cdot \boldsymbol{m}_2[r] &= (c+1) \cdot (k - \boldsymbol{m}_2[\mathsf{i}] - \boldsymbol{m}_2[\mathsf{f}]) - \boldsymbol{m}_2[p] && \text{(by (9))} \\ &= (c+1) \cdot (k - \boldsymbol{m}_1[\mathsf{f}]) - (k - \boldsymbol{m}[\mathsf{f}]) && \text{(by def. of } \boldsymbol{m}_2 \text{ and (8))} \\ &= (c+1) \cdot (k - \boldsymbol{m}[\mathsf{f}]) - (k - \boldsymbol{m}[\mathsf{f}]) && \text{(by def. of } \boldsymbol{m}_1) \\ &= c \cdot (k - \boldsymbol{m}[\mathsf{f}]). \end{aligned}$$

Altogether, we have $\boldsymbol{m}_2[r] = (k - \boldsymbol{m}[\mathsf{f}]) = \boldsymbol{m}_2[p]$. Thus, from $\boldsymbol{m}_2$, transition t_f can be fired $k - \boldsymbol{m}[\mathsf{f}]$ times, which leads to marking

$$\{\mathsf{f}\colon \boldsymbol{m}_1[\mathsf{f}] + (k - \boldsymbol{m}[\mathsf{f}])\} = \{\mathsf{f}\colon \boldsymbol{m}[\mathsf{f}] + k - \boldsymbol{m}[\mathsf{f}]\} = \{\mathsf{f}\colon k\}.$$

This concludes the proof of k-soundness as $\boldsymbol{m} \to^* \boldsymbol{m}_1 \to^* \boldsymbol{m}_2 \to^* \{\mathsf{f}\colon k\}$.

It remains to consider the case where $k = c$. We have

$$\{\mathsf{i}\colon c\} \to^{t_\mathsf{i}^c} \{p\colon (c+1) \cdot c\} \to^{t_r^{c+1}} \{r\colon (c+1)\}.$$

No transition is enabled in the latter marking. So, we have $\{r\colon (c+1)\} \not\to^* \{\mathsf{f}\colon c\}$ and hence c-unsoundness follows. We are done proving this item.

Item 4. Let $k \in \mathbb{N}_{\geq 1}$ be a number that is not a multiple of c. Let us first show that $\{\mathsf{i}\colon k\} \not\to^* \{\mathsf{f}\colon k\}$. For the sake of contradiction, assume there exists a run ρ such that $\{\mathsf{i}\colon k\} \to^\rho \{\mathsf{f}\colon k\}$. Note that ρ needs to fire t_i exactly k times, since no other transition consumes from i. Without loss of generality, let us reorder ρ into a run ρ' such that any firing of t_i happens at the beginning. Let us write $\rho' = t_\mathsf{i}^k \sigma$, where σ does not contain t_i. We have that $\{\mathsf{i}\colon 1\} \to^{t_\mathsf{i}^k} \{u\colon k, d\colon k\}$. The only transition consuming from u is t_u. Since k is not a multiple of c, and since t_u consumes c tokens from u, place u can never be emptied. Thus $\{u\colon k, d\colon k\} \not\to^* \{\mathsf{f}\colon 1\}$.

Next, let us show that $\{\mathsf{i}\colon mc\} \to^* \{\mathsf{f}\colon mc\}$ for any $m \in \mathbb{N}_{\geq 1}$. It follows from

$$\begin{aligned} \{\mathsf{i}\colon mc\} \to^{t_\mathsf{i}^{mc}} \{u\colon mc, d\colon mc\} \to^{t_d^{m(c-1)}} \{u\colon mc, d\colon m, \mathsf{f}\colon m(c-1)\} \\ \to^{t_u^m} \{\mathsf{f}\colon mc\}. \end{aligned}$$

Finally, we show that $N_{\neg\text{sound}}$ is not structurally sound. It suffices to show that it is mc-unsound for all $m \in \mathbb{N}_{\geq 1}$. Note that

$$\begin{aligned} \{\mathsf{i}\colon mc\} \to^{t_\mathsf{i}^{mc}} \{u\colon mc, d\colon mc\} \to^{t_u^m} \{d\colon (m-1)c, \mathsf{f}\colon m\} \\ \to^{t_d^{((m-1)c)-1}} \{d\colon 1, \mathsf{f}\colon mc-1\}. \end{aligned}$$

No transition is enabled in the latter marking, so mc-unsoundness follows. □