Soundness of Workflow Nets: Classification, Decidability, and Analysis

W.M.P. van der Aalst[1,2], K.M. van Hee[1], A.H.M. ter Hofstede[2], N. Sidorova[1], H.M.W. Verbeek[1], M. Voorhoeve[1], and M.T. Wynn[2]

[1] Department of Mathematics and Computer Science,
Eindhoven University of Technology
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands.
w.m.p.v.d.aalst@tue.nl
[2] Business Process Management Group, Queensland University of Technology
P.O. Box 2434, Brisbane Qld 4001, Australia.

Abstract. *Workflow nets*, a particular class of Petri nets, have become one of the standard ways to model and analyze workflows. Typically, they are used as an abstraction of the workflow that is used to check the so-called *soundness property*. This property guarantees the absence of livelocks, deadlocks, and other anomalies that can be detected without domain knowledge. Several authors have proposed alternative notions of soundness and have suggested to use more expressive languages, e.g., models with cancellations or priorities. This paper provides an *overview of the different notions of soundness and investigates these in the presence of different extensions of workflow nets*. We will show that the eight soundness notions described in the literature are decidable for workflow nets. However, most extensions will make all of these notions undecidable. These new results show the theoretical limits of workflow verification. Moreover, we discuss some of the analysis approaches described in the literature.

Keywords: Petri Nets, Decidability, Workflow Nets, Reset Nets, Soundness, and Verification.

1 Introduction

In the last 15 years, we have witnessed a shift from "data-aware" information systems to "process-aware" information systems [24]. To support business processes an enterprise information system needs to be aware of these processes and their organizational context. Early examples of process-aware information systems were called WorkFlow Management (WFM) systems [4, 32, 38, 46, 47, 51, 61, 76]. In more recent years, vendors prefer the term Business Process Management (BPM) systems. BPM systems have a wider scope than the classical WFM systems and are not just focusing on process automation. BPM systems tend to provide more support for various forms of analysis and management support. Both WFM and BPM aim to support operational processes that we refer to as "workflow processes" or simply "workflows".

The flow-oriented nature of workflow processes makes the Petri net formalism a natural candidate for the modeling and analysis of workflows. Most workflow management systems provide a graphical language which is close to Petri nets. Although the routing elements are different from Petri nets, the informal semantics of the languages used are typically token-based and hence a (partial) mapping is relatively straightforward. A characteristic of workflow processes is that they are typically case-oriented, i.e., processes can be instantiated for multiple cases but the life-cycles of different cases do not get intertwined. This can be illustrated using Figure 1. The process represented by the "cloud" can be instantiated by putting tokens on the input place *start*. Each of these tokens represents the creation of a particular case. The goal is that after a while there will be a token in output place *end* for each initiated case.

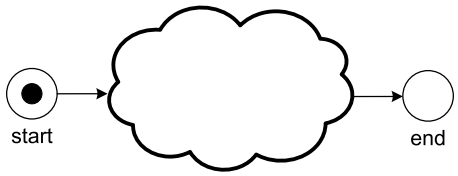


Fig. 1. A WF-net is a Petri net with a start and an end place. The goal is that a case initiated via place *start* successfully completes by putting a token in place *end*.

This paper focuses on processes having the structure shown in Figure 1. These are so-called *workflow nets* (WF-nets). WF-nets were introduced in [1, 2]. Together these two papers got more than one thousand references illustrating the interest in the topic.[3] In the context of WF-nets a correctness criterion called *soundness* has been defined [1, 2]. A WF-net such as the one sketched in Figure 1 is sound if and only if the following three requirements are satisfied: (1) *option to complete*: for each case it is always still possible to reach the state which just marks place *end*, (2) *proper completion*: if place *end* is marked all other places are empty for a given case, and (3) *no dead transitions*: it should be possible to execute an arbitrary activity by following the appropriate route through the WF-net. In [1, 2] it was shown that soundness is decidable and that it can be translated into a liveness and boundedness problem, i.e., a WF-net is sound if and only if the corresponding short-circuited net is live and bounded.

Since the mid-nineties many people have been looking at the verification of workflows. These papers all assume some underlying model (e.g., WF-nets) and some correctness criterion (e.g., soundness). Hence there are two dimensions when considering workflow verification:

[3] In fact, [2] is the second most cited workflow paper after [32] according to Google Scholar (visited on January 9th, 2008).

– *Expressiveness of the model.* Some authors assume a model that is less expressive than WF-nets, e.g., there are many variants of the so-called workflow graph model [66] which is essentially a free-choice net and thus easier to analyze than WF-nets without restrictions. Other authors propose more expressive models, e.g., models that allow for cancellation, priorities, data dependencies, recursion, and complex joins such as the inclusive OR-join [43, 79].
– *Correctness criterion.* The notion of soundness defined in [1, 2] is intuitively appealing. However, some authors suggest weakening the correctness notion [17–19, 52, 53, 64] while others propose to strengthen the correctness notion [36, 34, 70].

In this paper, we are interested in the verification of different variants of WF-nets using different soundness notions. We will systematically consider *four classes of WF-nets and eight notions of soundness* and focus on the decidability of the corresponding 4*8=32 verification problems.

The four classes of WF-nets are based on two possible extensions of WF-nets: *reset arcs* and *inhibitor arcs*. We will show that other extensions do not influence the expressive power of WF-nets and therefore are less relevant. Reset arcs are closely linked to notions of cancellation present in some of the more advanced languages. Inhibitor arcs allow for the modeling of advanced constructs such as priorities, preemption, OR-joins, etc.

The eight notions of soundness used in this paper have been identified after a thorough analysis of literature. Some of the notions weaken one or more of the requirements in the original definition [1, 2], e.g., the requirement that the net should not have dead transitions or the requirement that the net should always terminate properly. Other notions strengthen some of the conditions stated in [1, 2], e.g., the requirement that soundness still holds even if the net is instantiated multiple times in parallel.

Thus far the decidability of soundness has not been investigated systematically. In fact, as far as we know, *this is the first paper to investigate decidability of soundness for WF-nets with reset and or inhibitor arcs.*

The main motivation for this research is that researchers continue to come up with new workflow models and verification techniques. By providing a systematic overview, we hope to reveal the fundamental limits of workflow verification. This way we hope to avoid that authors continue to come up with verification problems and approaches that turn out to be special cases of already know results, i.e., we want to help the researchers with positioning their research problems as special cases of already known results, thus avoiding re-inventing the wheel. Moreover, we show that most correctness notions are in fact undecidable when combined with certain extensions that correspond to reset or inhibitor arcs. This clearly shows the theoretical limits of workflow verification.

The remainder of this paper is organized as follows. First, we briefly present an overview of related work (Section 2). A more detailed review of related work is given in later parts of the paper, e.g., when introducing the various soundness notions. Then, Section 3 presents some of the preliminaries (mathematical no-

tations and Petri net basics). Section 4 presents the basic notion of a WF-net and introduces the four classes of WF-nets investigated in this paper. In Section 5 the classical notion of soundness is introduced followed by definitions of seven other notions of soundness considered in the literature. Section 6 presents the main results. It systematically investigates the four classes of WF-nets and eight notions of soundness and focuses on the decidability of the corresponding 4*8=32 verification questions. Section 7 provides pointers to different analysis approaches. The goal is not to present new methods but to provide a high-level overview of existing approaches. Here we also emphasize that despite the fact that many verification questions are undecidable, a more pragmatic approach can help in finding numerous errors. Section 8 concludes the paper by summarizing the results and reflecting on the state-of-the-art in workflow verification.

2 Related Work

Since the mid nineties, many researchers have been working on workflow verification techniques [1–3, 5, 6, 8–11, 14, 15, 19, 21, 30, 31, 36, 37, 40, 45, 48–50, 53–59, 66–69, 71–74, 78–80]. It is impossible to give a complete overview here. Moreover, most of the papers on workflow verification focus on rather simple languages, e.g., AND/XOR-graphs which are even less expressive than classical Petri nets. Therefore, we only mention the work directly relevant to this paper.

The use of Petri nets in workflow verification has been studied extensively. In [1, 2] the foundational notions of WF-nets and soundness are introduced. In [35, 36] two alterative notions of soundness are introduced: k-soundness and generalized soundness. These notions allow for dead parts in the workflow but address problems related to multiple instantiation. In [52, 53] the notion of weak soundness is proposed. This notion allows for dead transitions. The notion of relaxed soundness is introduced in [17–19]. This notion allows for potential deadlocks and livelocks, however, for each transition there should be at least one proper execution. Lazy soundness [64, 63] is another variant that only focuses on the end place and allows for excess tokens in the rest of the net. Finally, the notions of up-to-k-soundness and easy soundness are introduced in [70]. More details on these notions proposed in the literature are given in Section 5.

Most soundness notions (except generalized soundness [35, 36]) can be investigated using classical model checking techniques that explore the state space. However, such approaches can be intractable or even impossible because the state-space may be infinite. Therefore, alternative approaches that avoid constructing the (full) state space have been proposed. [3] describes how structural properties of a workflow net can be used to detect the soundness property. [72, 73] presents an alternative approach for deciding relaxed soundness in the presence of OR-joins using invariants. The approach taken results in the approximation of OR-join semantics and transformation of YAWL nets [7] into Petri nets with inhibitor arcs. In the general area of reset nets, Dufourd et al.'s work has provided valuable insights into the decidability status of various properties of reset nets

including reachability, boundedness and coverability [22, 23, 29]. For decidability results for ordinary Petri nets we refer to [16, 25–27].

A number of authors have investigated reduction rules for Petri nets and for various subclasses of Petri nets. In Murata's paper [62], six reduction rules are presented for Petri nets and this set of rules can be used as a starting point for workflow reduction rules. In [20], a set of reduction rules is proposed for free-choice Petri nets while preserving well-formedness. Berthelot presents a set of reduction rules for general Petri nets [12, 13]. Reduction rules have been suggested to be used together with Petri nets for the verification of workflows (cf. Chapter 4 in [4]). Similar approaches have been applied to other languages such as EPCs [42, 43], BPMN [77], etc. Six reduction rules that preserve correctness for EPCs including reduction rules for trivial constructs, simple splits and joins, similar splits and joins, XOR loop and optional OR-loop are proposed in [21]. In [67, 68] a set of reductions rules for AND-XOR graphs (i.e., a special case of free-choice nets) is presented. The authors claim that these rules are complete (i.e., any correct workflow can be reduced completely). However, as shown in [5, 49] this is not the case. This can be easily corrected by using the reduction rules presented in [20] or by using a more direct method (e.g., based on the Rank Theorem as shown in [5]). None of the reduction rules mentioned above takes cancellation into account. This case is handled in [81]. In [75] the soundness preserving reduction rules are extended to nets with inhibitor arcs.

We would also like to refer to some empirical work on workflow verification. A detailed analysis of the SAP reference model is presented in [58, 60]. Here 604 EPC models [42, 43] are automatically translated to YAWL [7] and analyzed using Petri-net invariants. This study showed, using a simple technique such as invariants, that at least 5.6 percent of SAP's EPC models have obvious flaws (deadlocks, etc.). Later this study was extended to a larger set of models (more than 2000 EPC models from various sources) using more precise analysis techniques [57, 59]. Of these more than 2000 models at least 10 percent has errors. In [71] a set of 340 real business processes modeled with the IBM WebSphere Business Modeler is analyzed.

Empirical research clearly shows that modelers are likely to make errors if they are not supported by analysis tools. This illustrates the practical relevance of the research on workflow verification.

3 Preliminaries

This section introduces some of the basic mathematical and Petri-net related concepts used in the remainder of this paper.

3.1 Multi-sets, Sequences, and Matrices

Let A be a set. $\mathbb{B}(A) = A \to \mathbb{N}$ is the set of multi-sets (bags) over A, i.e., $X \in \mathbb{B}(A)$ is a multi-set where for each $a \in A$: $X(a)$ denotes the number of times a is included in the multi-set. The sum of two multi-sets $(X + Y)$, the

difference ($X - Y$), the presence of an element in a multi-set ($x \in X$), and the notion of sub-multi-set ($X \leq Y$) are defined in a straightforward way and they can handle a mixture of sets and multi-sets. The operators are also robust with respect to the domains of the multi-sets, i.e., even if X and Y are defined on different domains, $X + Y$, $X - Y$, and $X \leq Y$ are defined properly by taking the union of the domains where needed. $|X| = \sum_{a \in A} X(a)$ is the size of some multi-set X over A. $X(A') = \sum_{a \in A'} X(a)$ denotes the number of elements in X with a value in $A' \subseteq A$. $\pi_{A'}(X)$ is the projection of X onto $A' \subseteq A$, i.e., $(\pi_{A'}(X))(a) = X(a)$ if $a \in A'$ and $(\pi_{A'}(X))(a) = 0$ if $a \notin A'$.

To represent a concrete multi-set we use square brackets, e.g., $[a, a, b, a, b, c]$, $[a^3, b^2, c]$, and $3[a] + 2[b] + [c]$ all refer to the same multi-set with six elements: 3 a's, 2 b's, and one c. $[\,]$ refers to the empty bag, i.e., $|[\,]| = 0$.

Every multi-set over a finite domain can be represented as a vector, i.e., $X \in \mathbb{B}(A)$ can be represented as a row vector $(X(a_1), X(a_2), \ldots, X(a_n))$ where a_1, a_2, ... ,a_n enumerate the domain of X. $(X(a_1), X(a_2), \ldots, X(a_n))^T$ denotes the corresponding column vector (^{T} transposes the vector). Assume X is an $k \times \ell$ matrix, i.e., a matrix with k rows and ℓ columns. A row vector can be seen as $1 \times \ell$ matrix and a column vector can be seen as a $k \times 1$ vector. $X(i, j)$ is the value of the element in the i^{th} row and the j^{th} column. Let X be a $k \times \ell$ matrix and Y an $\ell \times m$ matrix. The product $X \cdot Y$ is the product of X and Y yielding a $k \times m$ matrix, where $(X \cdot Y)(i, j) = \sum_{1 \leq q \leq \ell} X(i, q)Y(q, j)$. The sum of two matrices having the same dimensions is denoted by $X + Y$, i.e., $(X + Y)(i, j) = X(i, j) + X(i, j)$.

For a given set A, A^* is the set of all finite sequences over A. A finite sequence over A of length n is a mapping $\sigma \in \{1, \ldots, n\} \rightarrow A$. Such a sequence is represented by a string, i.e., $\sigma = \langle a_1, a_2, \ldots, a_n \rangle$ where $a_i = \sigma(i)$ for $1 \leq i \leq n$. $hd(\sigma, k) = \langle a_1, a_2, \ldots, a_k \rangle$, i.e., the sequence of just the first k elements. Note that $hd(\sigma, 0)$ is the empty sequence.

For a relation R on A, i.e., $R \subseteq A \times A$, we define R^* as the reflexive transitive closure of R.

For any sequence $\sigma \in \{1, \ldots, n\} \rightarrow A$ over some finite set A, the Parikh vector $\overrightarrow{\sigma}$ maps every element a of A onto the number of occurrences of a in σ, i.e., $\overrightarrow{\sigma} \in \mathbb{B}(A)$ where for any $a \in A$: $\overrightarrow{\sigma}(a) = |\{i \in \mathbb{N} \mid 1 \leq i \leq n \ \wedge \ \sigma(i) = a\}|$. The projection of σ into some set $X \subseteq A$, notation $proj_X(\sigma)$, is the sequence obtained by removing all elements that are not in X.

3.2 Basic Petri nets

This subsection briefly introduces some basic *Petri net* terminology [20, 39, 65] and notations used in the remainder of this paper. First, we informally introduce the classical Petri net. In the next subsection, this model is extended and further formalized.

Definition 1 (Basic Petri net). *A basic Petri net is a triple (P, T, F). P is a finite set of places, T is a finite set of transitions ($P \cap T = \emptyset$), and $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation).*

Figure 2 shows a basic Petri net. Places are represented by circles and transitions are represented by squares.

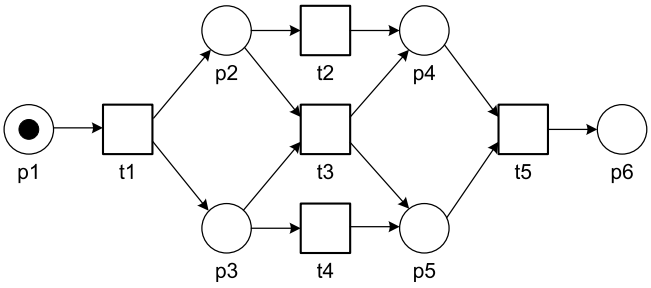


Fig. 2. A basic Petri net with places $\{p1, p2, p3, p4, p5, p6\}$ and transitions $\{t1, t2, t3, t4, t5\}$.

For any relation/directed graph $G \subseteq A \times A$ (including the graph defined by the flow relation of a Petri net F), we define the preset $\bullet a = \{a_1 \mid (a_1, a) \in G\}$ and postset $a\bullet = \{a_2 \mid (a, a_2) \in G\}$ for any node $a \in A$. We use $\overset{G}{\bullet}\, a$ or $a \overset{G}{\bullet}$ to explicitly indicate the context G if needed. Based on the flow relation F we use this notation as follows. $\bullet t$ denotes the set of input places for a transition t. The notations $t\bullet$, $\bullet p$ and $p\bullet$ have similar meanings, e.g., $p\bullet$ is the set of transitions sharing p as an input place. In the Petri net shown in Figure 2: $\bullet p5 = \{t3, t4\}$, $p5\bullet = \{t5\}$, $\bullet t3 = \{p2, p3\}$, $t3\bullet = \{p4, p5\}$, etc.

At any time a place contains zero or more *tokens*, drawn as black dots. The state of the Petri net, often referred to as *marking*, is the distribution of tokens over its places, i.e., $M \in \mathbb{B}(P)$. In the Petri net shown in Figure 2 only one place is initially marked ($p1$), i.e., $M = [p1]$. Note that more places could be marked in the initial state and that places can be marked with multiple tokens.

For a basic Petri net, we assume the standard *firing rule*, i.e., a transition t is said to be *enabled* with respect to some marking M if and only if each input place p of t contains at least one token. An enabled transition may *fire*, and if transition t fires, then t *consumes* one token from each input place p of t and *produces* one token for each output place p of t. For example, in Figure 2, $t1$ is enabled and the firing of $t1$ will result in the state that marks places $p2$ and $p3$. In this state $t2$, $t3$, and $t4$ are enabled. If $t2$ fires, $t3$ becomes disabled, but $t4$ remains enabled. Similarly, if $t4$ fires, $t3$ becomes disabled, but $t2$ remains enabled, etc.

In the next subsection, we will formalize the firing rule for an extended class of Petri nets. Before doing so, we introduce some well-known subclasses of Petri nets.

Definition 2 (Net classes). *Let $N = (P, T, F)$ be a basic Petri net.*

– N *is a* state machine net *if and only if* $\forall_{t \in T} \mid \bullet t| = |t \bullet | = 1$.
– N *is a* marked graph *if and only if* $\forall_{p \in T} \mid \bullet p| = |p \bullet | = 1$.
– N *is a* free-choice net *if and only if* $\forall_{t_1, t_2 \in T} (\bullet t_1 \cap \bullet t_2 \neq \emptyset) \Rightarrow (\bullet t_1 = \bullet t_2)$.

The Petri net shown in Figure 2 does not fit into any of the classes defined above. If we remove transition $t3$, the resulting net is free-choice. These different net classes are relevant from the viewpoint of analysis. For example, liveness and boundedness (two behavioral properties) can be decided in polynomial time for free-choice nets while this is not the case for non-free-choice nets [20].

A Petri net is connected if there is path from any node (place or transition) to any other node in the graph while ignoring the direction of the arcs, i.e., the Petri net cannot be partitioned in two disconnected parts. In the remainder we assume any Petri net to be connected and having at least two nodes, i.e., at least a place and a transition.

3.3 Extended Petri nets

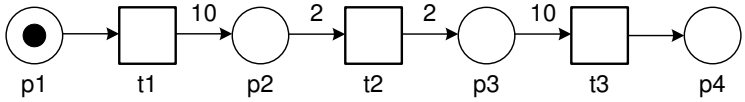
The basic Petri net model (Definition 1) is very simple and is not able to express all routing constructs one may encounter in real-life workflows. However, there are several obvious extensions of the basic model. Some of these extensions enhance the expressiveness (e.g., reset and inhibitor arcs) while other extensions only provide convenient shorthands (e.g., arc weights).

When modeling workflows in term of Petri nets, transitions correspond to activities. Let $\mathcal{A}$ be a universe of activity labels, i.e., $a \in \mathcal{A}$ refers to some activity. Multiple transitions can refer to the same activity, i.e., have the same activity label. The special label τ refers to a *silent step* [33]. We also say that transitions bearing the τ label are "invisible", i.e., transitions not corresponding to any activity and only added for routing purposes. Note that $\tau \notin \mathcal{A}$.

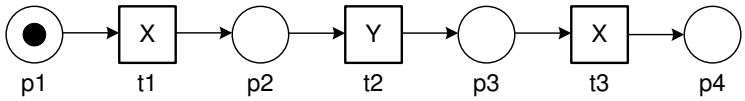
Definition 3 (Extended Petri net). *An extended Petri net is a tuple* (P, T, F, W, A, L, R, H)*, where:*

– (P, T, F) *is a basic Petri net,*
– $W \in F \rightarrow I\!N \setminus \{0\}$ *is an (arc) weight function,*
– $A \subseteq \mathcal{A}$ *is a set of (activity) labels,*
– $L \in T \rightarrow A \cup \{\tau\}$ *is a labeling function,*
– $R \in T \rightarrow 2^P$ *is a function defining reset arcs, and*
– $H \in T \rightarrow 2^P$ *is a function defining inhibitor arcs.*

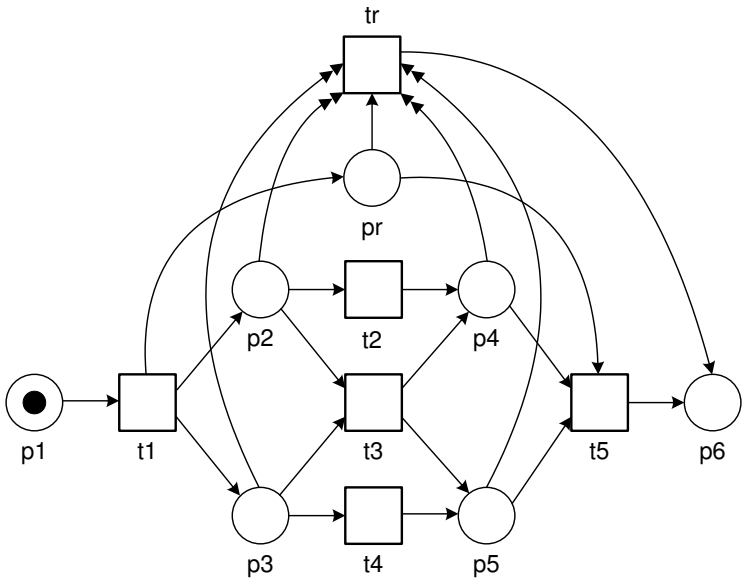
Figure 3 illustrates the four extensions mentioned in the above definition. Figure 3(a) shows the extension with arc weights. The arc from place $p1$ to transition $t1$ denotes an ordinary arc, i.e., $W(p1, t1) = 1$ indicating that $t1$ consumes one token from $p1$ when firing. The arc from transition $t1$ to place $p2$ has weight 10, i.e., $W(t1, p2) = 10$ indicating that $t1$ produces ten tokens for $p2$ when firing. We extend the weight function for the situation that there is not an arc connecting two nodes, i.e., $W(x, y) = 0$ if $(x, y) \notin F$. Moreover, for extended



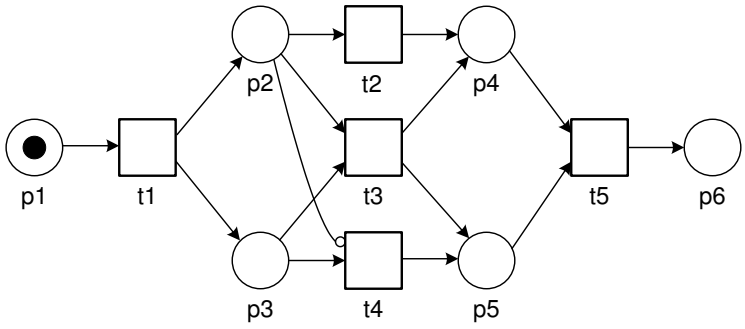
(a) Extended Petri net with arc weights



(b) Extended Petri net with transition labels



(c) Extended Petri net with reset arcs



(d) Extended Petri net with inhibitor arcs

Fig. 3. Four extended Petri nets illustrating the different extensions.

nets we redefine the preset and postset operator to return bags rather than sets: $\bullet a = [x^{W(x,y)} \mid (x,y) \in F \ \wedge \ a = y]$ and $a\bullet = [y^{W(x,y)} \mid (x,y) \in F \ \wedge \ a = x]$.

Figure 3(b) illustrates the notion of transition labels, i.e., each transition has a label. Note that multiple transitions may have the same label, e.g., $L(t1) = L(t3) = X$ in Figure 3(b). The label defines the "observable effect". In workflow terms: the label refers to the activity being executed while firing the corresponding transition. As a convention we will not show labels graphically if the labels coincide with transition identifiers, i.e., if $L(t) = t$, the label is omitted and just the transition identifier is shown (cf. Figure 3(a)).

The notion of reset arcs is illustrated in Figure 3(c). Here the four double-headed arcs are reset arcs. Note that $R(tr) = \{p2, p3, p4, p5\}$ and $R(t) = \emptyset$ for all other transitions t. Transition tr is enabled if and only if there is a token in place pr, i.e., reset arcs do not influence enabling. However, after the firing of tr all tokens are removed from the four places $p2$, $p3$, $p4$, and $p5$.

Figure 3(d) has one so-called inhibitor arc. The arc connecting $p2$ and $t4$ specifies that $p2$ should be empty when $t4$ fires. Note that $t4$ is enabled if and only if $p3$ contains at least one token and $p2$ contains *no* tokens. Note that in this example this implies that $t2$ has priority over $t4$, i.e., $t4$ can only occur after $t2$ has removed the token from $p2$. Note that $H(t4) = \{p2\}$ and $H(t) = \emptyset$ for all other transitions t.

After this informal introduction of the firing rule and the various extensions, we formalize this notion.

Definition 4 (Firing rule). *Let $N = (P, T, F, W, A, L, R, H)$ be an extended Petri net and $M \in \mathbb{B}(P)$ be a marking.*

- *A transition $t \in T$ is enabled, notation $(N, M)[t\rangle$, if and only if, $M \geq \bullet t$ and $M(H(t)) = 0$.*
- *An enabled transition t can fire while changing the state to M', notation $(N, M)[t\rangle(N, M')$, if and only if, $M' = \pi_{P \setminus R(t)}(M - \bullet t) + t\bullet$.*

The additional requirement $M(H(t)) = 0$ (i.e., $\sum_{p \in H(t)} M(p) = 0$) states that all places in $H(t)$ need to be empty for t to be enabled. Note that we use the notations introduced in Section 3.1 here. The resulting marking $M' = \pi_{P \setminus R(t)}(M - \bullet t) + t\bullet$ is obtained by first removing the tokens required for enabling: $M - \bullet t$. Then all tokens are removed from the reset places of t using projection. Applying function $\pi_{P \setminus R(t)}$ removes all tokens except the ones in the non-reset places $P \setminus R(t)$. Finally, the specified numbers of tokens are added to the output places. Note that $t\bullet$ is a *bag* of tokens.

$(N, M)[t\rangle(N, M')$ defines how a Petri net can move from one marking to another by firing a transition. We can extend this notion to firing sequences. Suppose $\sigma = \langle t_1, t_2, \ldots, t_n \rangle$ is a sequence of transitions present in some Petri net N with initial marking M. $(N, M)[\sigma\rangle(N, M')$ means that there is also a sequence of markings $\langle M_0, M_1, \ldots, M_n \rangle$ where $M_0 = M$, $M_n = M'$, and for any $0 \leq i < n$: $(N, M_i)[t_{i+1}\rangle(N, M_{i+1})$. Using this notation we define the set of reachable markings $R(N, M)$ as follows: $R(N, M) = \{M' \in \mathbb{B}(P) \mid \exists_\sigma (N, M)[\sigma\rangle(N, M')\}$.

Note that by definition $M \in R(N, M)$ because the initial marking M is trivially reachable via the empty sequence ($n = 0$).

The notions from Definition 4 can easily be lifted to the level of labels. $(N, M)[(a)\rangle$ means that starting in state M it is possible to reach a marking through a (possibly empty) sequence of silent transitions such that a transition t with a visible label a ($L(t) = a \in A$) becomes enabled. Note that a is enclosed by brackets to indicate that a is a label rather than a transition.. $(N, M)[(\sigma)\rangle(N, M')$ with $\sigma \in A^*$ means that there exists a sequence $\sigma' \in (A \cup \{\tau\})^*$ such that the projection of σ' onto visible transitions yields σ (i.e., $\sigma = proj_A(\sigma')$) and $(N, M)[\sigma\rangle(N, M')$.

For a marked Petri net we also define classical behavioral properties such as liveness and boundedness.

Definition 5 (Liveness, boundedness). *Let $N = (P, T, F, W, A, L, R, H)$ be an extended Petri net and $M \in \mathbb{B}(P)$ be a marking.*

- *(N, M) is live if and only if $\forall_{M' \in R(N,M)}\ \forall_{t \in T}\ \exists_{M'' \in R(N,M')}\ (N, M'')[t\rangle$.*
- *(N, M) is bounded if and only if $R(N, M)$ is finite.*
- *(N, M) is safe if and only if $\forall_{M' \in R(N,M)}\ \forall_{p \in P}\ M'(p) \leq 1$.*

A marked Petri net is live if from any reachable marking it is possible to (again) enable any transition. A place p is bounded if there is a $k \in \mathbb{N}$ such that $M'(p) \leq k$ for any reachable marking M'. A marked Petri net is bounded if all of its places are bounded. This is the case if and only if the number of reachable markings is finite. A net is safe if the number of tokens per place is bounded by 1, i.e., safeness is a special case of boundedness.

To conclude this section of preliminaries, we show the relation between the firing rule and the matrix representation of a Petri net.

Definition 6 (Incidence matrix). *Let $N = (P, T, F, W, A, L, R, H)$ be an extended Petri net. The incidence matrix of N, denoted $\tilde{N}$, is a $|P| \times |T|$ matrix with $\tilde{N}(p, t) = W(t, p) - W(p, t)$.*

The incidence matrix of a Petri net can be used for different types of analysis, e.g., based on $\tilde{N}$ it is possible to efficiently calculate place and transition invariants and to provide minimal (but not sufficient) requirements for the reachability of a marking [62]. For example, consider an extended Petri N net without reset arcs (i.e., $\forall_{t \in T}\ R(t) = \emptyset$) and let $M, M' \in \mathbb{B}(P)$ be two markings: if $(N, M)[\sigma\rangle(N, M')$, then by definition $M' = M + \tilde{N} \cdot \overrightarrow{\sigma}$ (i.e., the marking resulting after firing $\sigma \in T^*$ can be computed using a simple equation). Therefore, if there is no σ such that $M' = M + \tilde{N} \cdot \overrightarrow{\sigma}$, then M' is not reachable from M.

4 Workflow Nets

In the previous section, we considered arbitrary Petri nets without having an application in mind. However, when looking at workflows, we can make some assumptions about the structure of the Petri net. The idea of a workflow process

is that many *cases* (also called *process instances*) are handled in a uniform manner. The workflow definition describes the ordering of *activities* to be executed for each case. For example, the workflow for the handling of insurance claims describes how an individual claim is processed. Although at any point in time there may be many claims in the pipeline, the workflow definition only looks at one case in isolation. Note that cases may compete for resources. However, if one abstracts from resources, the cases are completely independent. Moreover, each of the cases will have a well-defined starting point and ending point. There is a point in time where the instance starts, i.e., the process is instantiated for a particular case, and, hopefully, there is a point in time where the instance is completed. Even if one considers multiple start activities and multiple end activities, from a conceptual viewpoint there is still a unique initial state and a unique final state.

These basic assumptions lead to the notion of a *WorkFlow net* (WF-net) [1, 2]. Using Figure 1 we already informally introduced the notion of such a WF-net and now it is time to formalize this notion.

Definition 7 (WF-net). *An extended Petri net* $N = (P, T, F, W, A, L, R, H)$ *is a WorkFlow net (WF-net) if and only if*

- *There is a single source place* i*, i.e.,* $\{p \in P \mid \bullet p = \emptyset\} = \{i\}$.
- *There is a single sink place* o*, i.e.,* $\{p \in P \mid p\bullet = \emptyset\} = \{o\}$.
- *Every node is on a path from* i *to* o*, i.e., for any* $n \in P \cup T$*:* $(i, n) \in F^*$ *and* $(n, o) \in F^*$.
- *There is no reset arc connected to the sink place, i.e.,* $\forall_{t \in T}\ o \notin R(t)$.

Figures 2 and 3 show five WF-nets. In each of these nets $i = p1$ is the source place and either $p4$ or $p6$ is the sink place o. Every node in each of these Petri nets is on a path from i to o. The requirement that $\forall_{t \in T}\ o \notin R(t)$ has been added to emphasize that termination should be irreversible, i.e., it is not allowed to complete (put a token in o) and then undo this completion (remove the token from o).

Transitions in a WF-net correspond to activities. Note that Definition 7 allows for multiple start ($i\bullet$) and end ($\bullet o$) activities. It is also easy to generalize this definition to multiple start and end places. However, to simplify notation we assume a single start and end place. Definition 7 also does not explicitly address notions such as AND/XOR-splits/joins. By default a transition corresponds to an AND-join/AND-split activity. To model an XOR-join or XOR-split activity, the desired behavior can be added using silent transitions (transitions with label τ) or duplicate transitions (multiple transitions having identical labels). Such transformations are trivial as shown in [2].

Definition 7 focuses on the control-flow and abstracts from resources, interaction, and data. While the definition can be extended with additional perspectives, we deliberately abstract from such things for several reasons. First of all, WF-nets are *not* proposed as an end-user language. We envision that people use languages such as (extended) EPCs [43], BPMN [77], etc. or some proprietary workflow language. The control-flow aspects of such languages can

be mapped onto WF-nets. We aim to investigate the foundations of workflow modeling and analysis and do not want to focus on a particular language. Resources, interaction, and data can be added by introducing resource places (cf. resource-constrained WF-nets [9, 34]), communication places (cf. open WF-nets [50, 55, 56]), or data places. However, for a more realistic modeling of these perspectives one needs to resort to colored Petri nets [39]. As a result, analysis (other than simulation) tends to get intractable. Moreover, it may be impossible to accurately model these perspectives. For example, a decision may be based on some complex calculation, external data, or human judgment. Hence, many decisions need to be modeled as non-deterministic choices anyway. Therefore, it seems natural to abstract from these aspects and first analyze the control-flow in isolation as further motivated in [2] and many other papers on workflow verification.

5 Soundness

Based on the notion of WF-nets we now investigate the fundamental question: "Is the workflow correct?". If one has domain knowledge, this question can be answered in many different ways. However, without domain knowledge one can only resort to generic questions such as: "Does the workflow terminate?", "Are there any deadlocks?", "Is it possible to execute activity A?", etc. Such kinds of generic questions triggered the definition of *soundness* [1, 2]. In this paper, we consider different soundness notions. However, we first start with the original definition given in [1].

Definition 8 (Classical soundness [1, 2]). *Let $N = (P, T, F, W, A, L, R, H)$ be a WF-net. N is sound if and only if the following three requirements are satisfied:*

- *Option to complete:* $\forall_{M \in R(N,[i])}\ [o] \in R(N, M)$.
- *Proper completion:* $\forall_{M \in R(N,[i])}\ (M \geq [o]) \Rightarrow (M = [o])$.
- *No dead transitions:* $\forall_{t \in T}\ \exists_{M \in R(N,[i])}\ (N, M)[t\rangle$.

Figure 1 was used to informally introduce the notion of soundness. Note that here $i = start$ and $o = end$. Each of the five WF-nets depicted in figures 2 and 3 is sound.

The first requirement in Definition 8 states that starting from the initial state (just a token in place i), it is always possible to reach the state with one token in place o (state $[o]$). If we assume a strong notion of fairness, then the first requirement implies that eventually state $[o]$ is reached. Strong fairness, sometimes also referred to as "impartial" or "recurrent" [44], means that in every infinite firing sequence, each transition fires infinitely often. Note that weaker notions of fairness are not sufficient, see Figure 2 in [44]. However, such a fairness assumption is reasonable in the context of workflow management since all choices are made (implicitly or explicitly) by applications, humans or external actors. If we required termination without this assumption, all nets allowing loops in their

execution sequences would be called unsound, which is clearly not desirable. The second requirement states that the moment a token is put in place o, all the other places should be empty. The last requirement states that there are no dead transitions (tasks) in the initial state $[i]$.

By carefully looking at Definition 8 one can see that the second requirement is implied by the first one.

Proposition 1 (Proper completion is implied). *Let N be a WF-net. The "option to complete" implies "proper completion", i.e., $(\forall_{M \in R(N,[i])}\ [o] \in R(N,M)) \Rightarrow\ (\forall_{M \in R(N,[i])}\ (M \geq [o]) \Rightarrow (M = [o]))$.*

Proof. Once a token is put in o there is no way of removing it, because o is a sink place and reset arcs are not allowed to empty o. Any transition produces tokens. Hence, if there are at least two tokens of which one is in o (i.e., improper completion), then there will always be at least two tokens (i.e., no option to complete). □

Hence we can ignore the second requirement in Definition 8. The reason that we include it anyway is because it represents an intuitive behavioral requirement.

As pointed out in [1, 2], classical soundness of a WF-net without reset and/or inhibitor corresponds to liveness and boundedness of the so-called short-circuited net. The short-circuited net is the Petri net obtained by connecting o to i, thus making the net cyclic.

Lemma 1 (Soundness, liveness and boundedness). *Let N be a WF-net and $\overline{N}$ the extended Petri net obtained by connecting o to i through a new transition t^*.*

- *If N is sound, then $(\overline{N}, [i])$ is live.*
- *If N has no reset and inhibitor arcs, then N is sound if and only if $(\overline{N}, [i])$ is live and bounded.*

Proof. If N is sound, then the moment a token is put into o all other places are empty. Hence, t^* can only bring the net back to the initial state. Moreover, the set of reachable states of does not change by adding t^* and all transitions remain non-dead. Since $(\overline{N}, [i])$ can return to the initial state again and again, the net is live.

If N has no reset and inhibitor arcs, then the results presented in [2] apply. Hence soundness coincides with liveness and boundedness of the short-circuited net. □

Note that if N is sound, the net does not need to be bounded. For example, there could be a reset arc removing an arbitrary (i.e., unbounded) number of tokens before producing a token for o.

After the initial paper on soundness of WF-nets [1, 2] many other papers followed. Some extend the results while others explore alternative notions of soundness. In the remainder of this section we define seven alternative notions

described in the literature. These notions strengthen or weaken some of the requirements mentioned in Definition 8.

The first notion of soundness focuses on the "option to complete", i.e., the first requirement in Definition 8. Moreover, this notion is parameterized with a variable k which indicates the initial number of tokens in the source place.

Definition 9 (k-soundness [35, 36]). *Let N be a WF-net. N is k-sound if and only if $\forall_{M \in R(N,[i^k])}\ [o^k] \in R(N, M)$.*

Note that any WF-net is 0-sound and that 1-soundness corresponds to the first requirement in Definition 8. In Lemma 11 in [36] it is shown that also for k-soundness the "option to complete" implies "proper completion", i.e., Proposition 1 also holds for k tokens. We introduce k-soundness mainly to be able to define useful notions such as weak soundness and generalized soundness.

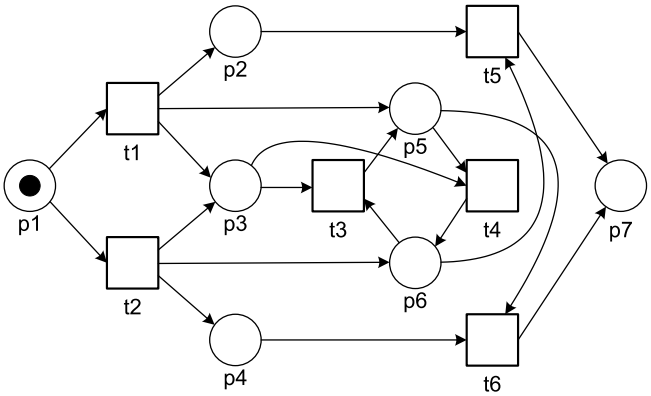


Fig. 4. A WF-net that is 1-sound but not 2-sound.

Consider the WF-net shown in Figure 4. This WF-net is classical sound and therefore also 1-sound. From the marking $[p1]$ there are two firing sequences both leading to $[o]$: $\langle t1, t4, t5\rangle$ and $\langle t2, t3, t6\rangle$. However, the net is not 2-sound, e.g., the sequence $\langle t1, t1, t4, t3\rangle$ results in a deadlock not being the desired final state. In fact it is not sound for any $k \geq 2$.

The notion of 1-soundness is also known as weak soundness [52, 53].

Definition 10 (Weak soundness [52, 53]). *Let N be a WF-net. N is weak sound if and only if N is 1-sound.*

Figure 5 shows three WF-nets that are all weak sound. The WF-net shown in Figure 5(a) is not classical sound, because transition $t3$ is dead. It is also not 2-sound because it is possible to reach state $[p3]$ when starting in $[p1^2]$ (i.e., a token is missing in the final state). Figure 5(b) is not 2-sound because of a

similar problem (state $[p3]$ is reachable from $[p1^2]$). Figure 5(c) is not 2-sound because the workflow may deadlock in state $[p1, p3]$. It is easy to see that similar problems occur when $k > 2$.

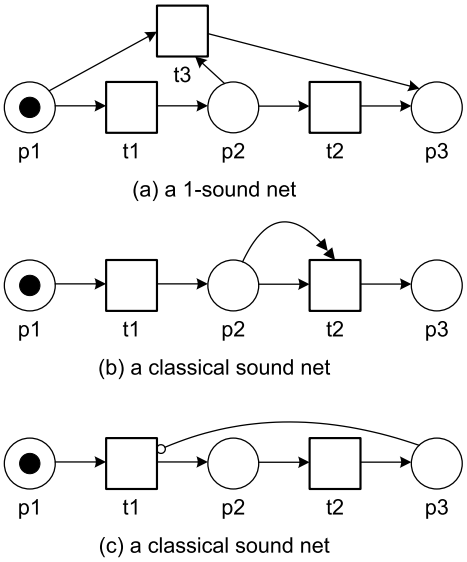


Fig. 5. Some more WF-nets that are 1-sound but not k-sound for any $k \geq 2$.

For a given k, it is easy to construct a WF-net that is k-sound but not $k+1$-sound. Therefore, we define the notion of up-to-k-soundness.

Definition 11 (up-to-k-soundness [70]). *Let N be a WF-net. N is up-to-k-sound if and only if N is l-sound for all $0 \leq l \leq k$*

Generalized soundness [35, 36] intuitively corresponds to up-to-∞-soundness, i.e., k-sound for any $k \in \mathbb{N}$.

Definition 12 (Generalized soundness [35, 36]). *Let N be a WF-net. N is generalized sound if and only if for all $k \in \mathbb{N}$: N is k-sound.*

The soundness notions discussed so far consider all possible execution paths and if for one path the desired end state is not reachable, the net is not sound. In a way this implies that the workflow is "lunacy proof", e.g., the user cannot select a path that will deadlock. The notion of relaxed soundness assumes a responsible user or environment, i.e., the net does not have to be "lunacy proof" as long as there exist "good" execution paths.

Definition 13 (Relaxed soundness [17–19]). *Let N be a WF-net. N is relaxed sound if and only if for each transition $t \in T$:*
$\exists_{M,M' \in R(N,[i])} \ (N, M)[t\rangle(N, M') \ \wedge \ [o] \in R(N, M')$.

Figure 6(a) shows a net that is not weak sound but that is relaxed sound. Note that firing sequences $\langle t2, t4, t6\rangle$ and $\langle t1, t3, t4, t5\rangle$ represent "good" executions ending in $[p6]$ and covering all transitions. It is assumed that the incorrect firing sequence $\langle t1, t3, t4, t6\rangle$ is avoided. As shown in [17, 18] it is possible to automatically convert a relaxed sound WF-net into a sound WF-net by blocking the undesired paths, i.e., the workflow engine can select the "good" behavior. The relaxed sound WF-net in Figure 6(a) can be converted into a sound WF-net by adding a place connecting $t2$ and $t6$.

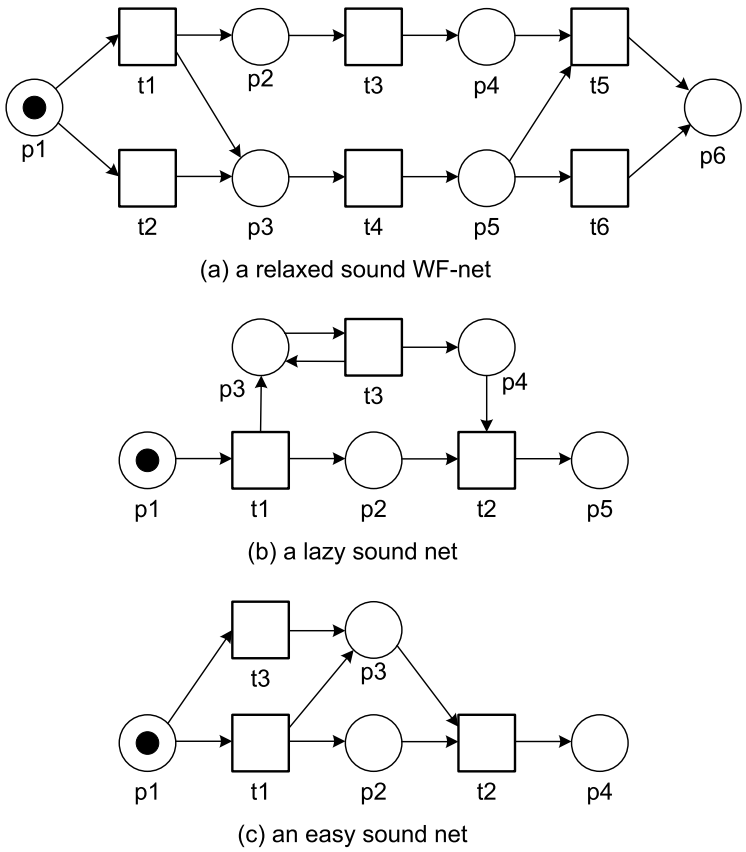


Fig. 6. Some more WF-nets illustrating relaxed, lazy, and easy soundness.

The soundness notions discussed so far focus on ending in a state with no tokens in any place other than the sink place. Lazy soundness weakens this requirement, i.e., tokens may be left behind as long as the sink place is marked precisely once.

Definition 14 (Lazy soundness [64, 63]). *Let N be a WF-net. N is lazy sound if and only if the following two requirements are satisfied:*

- *Option to complete:* $\forall_{M \in R(N,[i])}\ \exists_{M' \in R(N,M)}\ M'(o) = 1$.
- *Proper completion:* $\forall_{M \in R(N,[i])}\ M(o) \leq 1$.

The net in Figure 6(b) is lazy sound. Note that $t3$ can even fire repeatedly after putting a token in $p5$. The last notion of soundness, named easy soundness, only considers the possibility of the option to complete.

Definition 15 (Easy soundness [70]). *Let N be a WF-net. N is easy sound if and only* $[o] \in R(N, [i])$.[4]

Figure 6(c) is easy sound but does not satisfy any of the other 7 soundness notions. It is easy sound because the firing sequence $\langle t1, t2 \rangle$ indeed leads from $[i]$ to $[o]$.

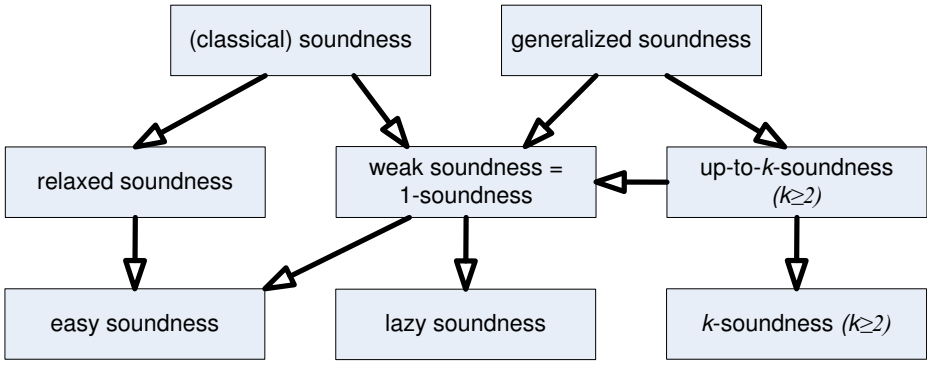


Fig. 7. Relationships between different various kinds of soundness ($\rightarrow$ = "implies").

While introducing the various soundness notions, we already indicated that some soundness notions are stronger than other soundness notions. Figure 7 shows the different implications. Classical soundness implies relaxed soundness and weak soundness. Weak soundness corresponds to the first requirement of classical soundness. Relaxed soundness corresponds to a weakening of this first requirement. It is trivial to see that generalized soundness implies weak soundness and up-to-k soundness. Up-to-k soundness of course also implies k soundness. Weak soundness implies lazy soundness because it weakens the "option to complete" and "proper completion" requirements by just considering sink place o. Easy soundness is implied by relaxed soundness and by weak soundness since

[4] In [70] this notion was named weak soundness but interpreted differently from [52, 53]. Hence, in this paper this notion is referred to as "easy" soundness to avoid confusion.

both imply that there is at least one path from $[i]$ to $[o]$. Note that Figure 7 only shows the transitive reduction of all implications, e.g., because of transitivity generalized soundness also implies lazy soundness.

The eight soundness notions mentioned in Figure 7 will be considered in the remainder of this paper. Several other soundness notions have been defined in the literature. For example, in the context of open WF-nets [50, 55, 56], cross-organizational WF-nets [45], and interacting BPMN/π-calculus processes [63] additional notions of soundness have been defined. However, these notions do not look at a single WF-net in isolation and therefore are outside the scope of this paper. Note that when considering interaction and/or resources different notions are needed.

6 Decidability

In this section we explore the different notions of soundness and the various classes of WF-nets and their decidability. Here will will focus on extended WF-nets and the eight notions of soundness defined earlier. First, we show that arc weights and transition labels are not relevant for decidability. Then, we show that WF-nets with inhibitor arcs are more expressive than nets with reset arcs. Based on these initial insights we present our decidability results.

6.1 Removing arc weights and labels

This subsection shows that arc weights and transition labels are not relevant for decidability. It is easy to see that transition labels do not influence soundness. Moreover, as Figure 8 shows, weighted arcs can also be removed. In Figure 8 it is assumed that k is the maximal arc weight on the input side and output side of p. Note that there happens to be an input transition producing k tokens and an output transition consuming k tokens. In the general case there may be arbitrarily many input and output transitions as long as they consume/produce not more than k tokens from/for place p. Suppose there is an output transition which consumes $l \leq k$ tokens from p in Figure 8(a). In Figure 8(b) this transition will have l out of the k newly added places as input. Any subset of these places will do, e.g., $\{p_1, \ldots, p_l\}$.

Note that the construction does not introduce any new type of arcs, e.g., inhibitor arcs are only needed if they were already present in the original net. Moreover, if the initial net has a WF-net structure, the resulting net also has this structure.

Since we can abstract from arc weights and transition labels, we restrict ourselves to *core WF-nets* in the remainder.

Definition 16 (Core WF-nets). *A core WF-net is a WF-net* (P, T, F, W, A, L, R, H) *where* $A = T$*, for all* $t \in T$*:* $L(t) = t$*, and for all* $f \in F$*:* $W(f) = 1$*. A core WF-net can be represented by* (P, T, F, R, H)*.*

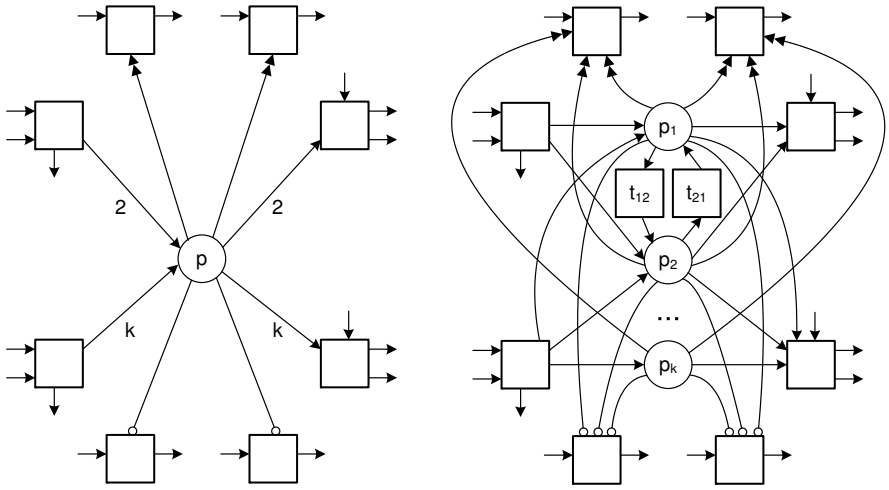


Fig. 8. Removing arc weights: place p is split up into k places and transitions are added such that tokens can freely move among these places. The arc weights 2 and k can be replaced by any weight $1 \leq l \leq k$ and the construction also works for more/less input/output transitions.

Since arc weights and transition labels are just "syntactical sugaring", we do not need to consider them when investigating soundness. For example, if generalized soundness is decidable for WF-nets without arc weights, then it will also be decidable for WF-nets with arc weights.

6.2 Inhibitor arcs can emulate reset arcs

Reset and inhibitor arcs are clearly more than "syntactical sugaring" and really add to the expressiveness, i.e., one can construct WF-nets with reset or inhibitor arcs that do not have an equivalent WF-net without such arcs.[5] However, as shown below, WF-nets with reset arcs can be translated into equivalent WF-nets with inhibitor arcs. First, we show this for arbitrary extended Petri nets.

Proposition 2 (Inhibitor arcs can emulate reset arcs). *Let N be an extended Petri net. All reset arcs can be replaced by inhibitor arcs without changing the behavior (modulo branching bisimulation [33]).*

Proof. It is easy to see that the construction shown in Figure 9 can be used to remove any reset arc. Transition t is replaced by a start transition t_s and an

[5] Note that the term "equivalent" is ill defined in this context. Although this statement is not very sensitive to the notion of equivalence considered, one can think of standard equivalence notions such as branching bisimulation [33].

end transition t_e. The start transition t_s has the same label as transition t and the end transition t_e has a τ label. Each transition with a reset arc is replaced by a small network as shown in Figure 9. Place x is added to guarantee mutual exclusion, i.e., there is one such place and all original transitions in the new net consume a token from x and return a token to x while all new start transitions consume a token from x and all new end transitions produce a token for x. As a result, any firing of t_s is followed by zero of more firings of t_c, followed by one firing of t_e. The effect of firing in one sequence $t_s(t_c)^n t_e$ is equivalent to firing t in the original. It is easy to establish a branching bisimulation relation between both nets by associating the label of t to t_s and giving transitions t_c and t_e a τ label. □

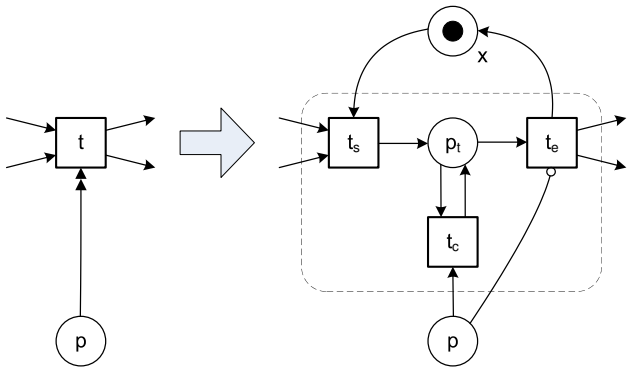


Fig. 9. Construction showing that reset arcs can be translated into inhibitor arcs.

Proposition 2 is also applicable to (core) WF-nets, i.e., by adding a new global start and end transition, place x can be marked and unmarked and the "WF-net structure" remains intact.

In the remainder of this section we investigate decidability for the eight notions of soundness and WF-nets with reset and/or inhibitor arcs. Here, we often use the following corollary.

Corollary 1 (Undecidability of weak soundness). *If a particular soundness property (classical soundness, k-soundness, weak soundness, etc.) is undecidable for WF-nets with reset arcs it will also be undecidable for WF-nets with inhibitor arcs.*

This corollary follows directly from Proposition 2. Any WF-net with reset arcs can be converted into a Petri net without reset arcs and just inhibitor arcs. Note that the resulting net is not a WF-net because of the initially marked x place. However, by adding a new "start transition" and a new "end transition", the net can be transformed into an equivalent WF-net.

6.3 Classical soundness

In this subsection, we explore the decidability of soundness for WF-nets. If a WF-net has no reset and no inhibitor arcs, we can use Lemma 1 to show that soundness is decidable. Such a WF-net N is sound if and only if $(\overline{N}, [i])$ is live and bounded. Since liveness and boundedness are both decidable, soundness is also decidable. For some subclasses (e.g., free-choice nets), this is even decidable in polynomial time [1, 2].

Unfortunately, soundness is not decidable for WF-nets with reset and/or inhibitor arcs. First, we show that reset arcs make the verification problem undecidable.

Theorem 1 (Undecidability of soundness). *Soundness is undecidable for WF-nets with reset arcs.*

Proof. Let (N, M_I) be an arbitrary marked Petri net with reset arcs but no inhibitors. In the general case it is known that reachability is undecidable for reset nets [22, 23]. Without loss of generality we can assume that N is connected and that every transition has input and output places, since any reset net can be translated into a behaviorally equivalent net that has these properties. Moreover, since coverability is decidable for reset nets [22, 29], we can assume that all dead transitions have been removed. (Because we can check whether $\bullet t$ is coverable from the initial marking, we can test whether transition t is dead for any $t \in T$.) Hence we may assume that (N, M_I) is connected, every transition has input and output places, and there are no dead transitions.

To show that soundness is undecidable, we construct a new net $(N', [i])$ which embeds (N, M_I) such that N' is sound if and only if some marking M_X is *NOT* reachable from (N, M_I). By doing so, we show that reachability in an arbitrary reset net can be analyzed through soundness, making soundness also undecidable.

The construction is shown in Figure 10. However, to explain this we first need to introduce some notation. P is the set of places in N and T is the set of transitions in N. Assume $\{i, o, u, s, v, w\} \cap P = \emptyset$ and $(\{a, b, c, z\} \cup \{z_p \mid p \in P\}) \cap T = \emptyset$. These are the "fresh" identifiers corresponding to the places and transitions added to N to form N'. $I \subseteq P$ are all the places that are initially marked in (N, M_I) and $X \subseteq P$ are the places that are marked in (N, M_X). As Figure 10 shows, transition c initializes the places in I, i.e., for $p \in I$: $W(c, p) = M_I(p)$.[6] Similarly, transition b can fire and consume all tokens from X if marking M_X is reached, i.e., for $p \in X$: $W(p, b) = M_X(p)$, and transition a marks the places in X appropriately, i.e., for $p \in X$: $W(a, p) = M_X(p)$. The transitions z and z_p ($p \in P$) have reset arcs from all places in N' except the new sink place o. Any transition in the original net has a bidirectional arc with s. All other connections are as shown in Figure 10.

The constructed net $(N', [i])$ has the following behavior. First a fires, marking u, v and the places in X. No transition $t \in T$ can fire because s is still empty

[6] Note that we are assuming weighted arcs here. However, as shown before these can be removed using the construction in Figure 8.

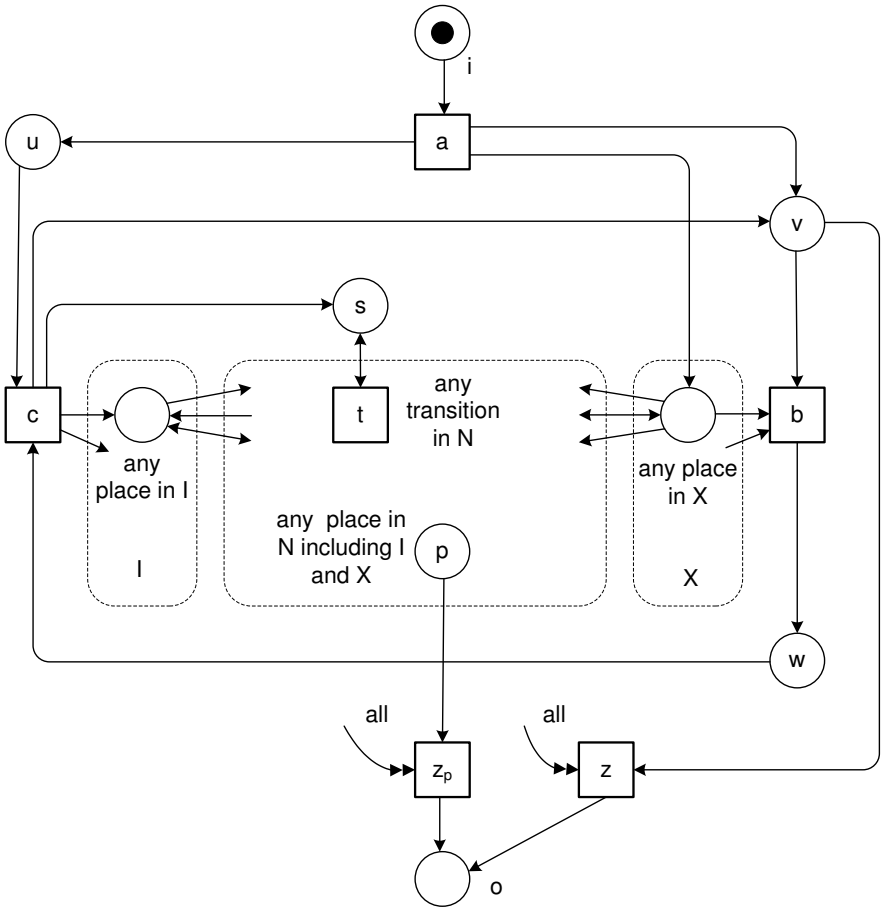


Fig. 10. Construction showing that soundness is undecidable for WF-nets with reset arcs. The original net comprises the three dashed areas: I is the set of places of N initially marked, X is the set of places that are marked in M_X, and all other nodes of N are shown in the dashed area in the middle. Note that I and X may overlap.

and c is also blocked because w is empty. The only two transitions that can fire are b and z. If z occurs, the net ends in marking $[o]$. If b fires, it will be followed by c. The firing of c brings the net into marking $M_I + [s, v]$. Note that in marking $M_I + [s, v]$ the original transitions are not constrained in any way and the embedded subnet can evolve as in (N, M_I) until one of the newly added transitions fires. Transitions $\{z_p \mid p \in P\}$ can fire as long as there is at least one token in a place in P and z can fire as long as there is a token in v. The firing of such a transition always leads to $[o]$, i.e., firing a transition in $\{z\} \cup \{z_p \mid p \in P\}$ always leads to the proper end state. Transition b can fire as soon as the embedded subnet has a marking which covers M_X.

It is obvious that net N' shown in Figure 10 is a WF-net, i.e., there is one source place i, one sink place o, all nodes are on a path from i to o, and there is no reset on o.

Now we can show that N' is sound if and only if some marking M_X is *NOT* reachable from (N, M_I):

- Assume marking M_X is reachable from (N, M_I). This implies that from $(N', [i])$ the marking $M_X + [s, v]$ is reachable. Hence b can fire for the second time resulting in a state $[s, w]$. In this state all transitions in T are blocked because transitions have input places and all input places in P are empty. Also all added transitions are dead in $[s, w]$. Hence a deadlock state $[s, w]$ is reachable from $(N', [i])$ implying that N' is not sound.
- Assume marking M_X is not reachable from (N, M_I) and M_X is also not coverable. This implies that b cannot fire for the second time. Hence, there always remain tokens in some place of P after initialization and it is always possible to terminate in state $[o]$ by firing one of the "z transitions". Moreover, none of the transitions is dead in $(N', [i])$ because $\{a, b, c, z\} \cup \{z_p \mid p \in P\}$ can fire and the transitions in T are not dead in (N, M_I) (because of the initial cleaning). Therefore, N' is indeed sound.
- Assume marking M_X is not reachable from (N, M_I) but M_X is coverable. This implies that in the embedded subnet it is only possible to reach states M' that are not covering M_X or that are bigger than M_X, i.e., $M' \geq M_X$ implies $M' \neq M_X$. For states smaller than M_X we have shown that soundness is not jeopardized. For states bigger than M_X, b can fire. However, if b fires, tokens remain in P and b cannot fire anymore. Hence, at least one transition in $\{z_p \mid p \in P\}$ is enabled at any time because one of the places in P is marked. As a result, it is always possible to terminate in state $[o]$ and N' is indeed sound.

Hence, if soundness is decidable for reset nets, then reachability is also decidable. This leads to a contradiction. Hence soundness is not decidable. □

Theorem 1 shows that the ability of cancellation combined with unbounded places makes soundness undecidable. This is a relevant result because many workflow languages have such features.

Since inhibitor arcs can emulate reset arcs (cf. Proposition 2), the undecidability result also applies to WF-nets with inhibitor arcs.

6.4 Weak soundness

Next, we investigate the decidability of weak soundness, also known as 1-soundness. Weak soundness corresponds to the first requirement of classical soundness. Since the second requirement is implied by the first one, the only difference is the third requirement, i.e., for weak soundness it is not required that there are no dead transitions. From the viewpoint of decidability this is less relevant because dead transitions can be removed from a WF-net with reset arcs.

Corollary 2 (Undecidability of weak soundness). *Weak soundness is undecidable for WF-nets with reset arcs.*

Proof. Let N be an arbitrary WF-net with reset arcs but no inhibitor arcs. Remove all dead transitions in $(N, [i])$ and let N' be the resulting WF-net. This is possible because coverability is decidable for reset nets [22, 29] and therefore it is possible to check for each transition t whether $\bullet t$ is coverable from the initial marking (see also proof of Theorem 1). Now N is weak sound if and only if N' is sound. Since Theorem 1 shows that soundness is undecidable, weak soundness is also undecidable. □

If a WF-net has no reset and no inhibitor arcs, weak soundness is decidable. Note that we can first remove all dead transitions from N, then soundness corresponds again to liveness and boundedness of the short-circuited net, which is decidable [1, 2].[7] It is also obvious that, in this case, weak soundness can be checked by simply inspecting the well-known coverability graph [28, 62].[8]

Because of Corollary 1, soundness is also undecidable for WF-nets with inhibitor arcs.

6.5 *k*-soundness

Now we consider the situation of k-soundness with $k \geq 2$. Just like 1-soundness, the decidability results are equal to classical soundness. The question whether a net is weak sound can be translated into a k-soundness question, and vice versa, for any k as shown in Figure 11.

Corollary 3 (Undecidability of k-soundness). *For any $k \geq 2$: k-soundness is undecidable for WF-nets with reset arcs.*

Proof. We will show that decidability of k-soundness for WF-nets with reset arcs would imply decidability of weak soundness for WF-nets with reset arcs, which would contradict Theorem 1. Let N be an arbitrary WF-net with reset arcs. From N we construct the WF-net N_k which embeds N and has parameter $k \geq 2$ as shown in Figure 11. Clearly, N_k is a WF-net. Again the arc weights can

[7] The removal of dead transitions could potentially transform a WF-net into a non-WF-net. However, this can be repaired easily by adding a new source place i_s, start transition t_s, sink place o_e, end transition t_e, and self-loop place x. t_s consumes a token from the new source place i_s and produces a token for the old source place i and the self-loop place x. t_e consumes a token from the old sink place o and x, and produces a token for the new sink place o_e. All original transitions have x as a self loop, i.e., they consume and produce a token from x.

[8] Note that given a Petri net there may be multiple coverability graphs, i.e., the classical Karp and Miller algorithm [41] does not necessarily produce a unique graph. This is not a problem, because the different graphs lead to identical conclusions. Moreover, as shown in [28] it is possible to construct a unique "minimal coverability graph". Therefore, we will refer to "the" coverability graph rather than "a" coverability graph in the context of a particular marked net.

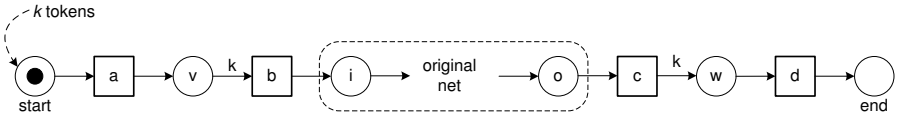


Fig. 11. The original WF-net is weak sound if and only if the constructed WF-net is k-sound. Because k tokens are put in place *start*, transition b fires once to initialize the original net. If the original net puts one token in o, then c can fire once followed by k executions of b, thus producing k tokens for place *end*.

be removed as shown in Figure 8; this is the reason for adding the additional transitions at the beginning and end. Moreover, it is easy to verify that N is weak sound if and only if N_k is k-sound. Therefore, we can apply Theorem 1 to show undecidability. □

Note that k-soundness is also undecidable for WF-nets with inhibitor arcs (cf. Corollary 1), but is decidable for WF-nets without reset and/or inhibitor arcs.

6.6 Up-to-k-soundness

Up-to-k-soundness can be translated into k "l-soundness" problems where $1 \leq l \leq k$. Therefore, intuitively it is no surprise that the results are identical.

Corollary 4 (Undecidability of up-to-k-soundness). *For any $k \geq 2$: up-to-k-soundness is undecidable for WF-nets with reset arcs.*

Proof. We will show that decidability of up-to-k-soundness for WF-nets with reset arcs would imply decidability of weak soundness for WF-nets with reset arcs, which would contradict Theorem 1. Let N be an arbitrary WF-net with reset arcs. From N we construct the WF-net N_k which embeds N and has parameter $k \geq 2$ as shown in Figure 12. This net is identical to the one in Figure 11 apart from the "bypass" transition x. N is weak sound if and only if

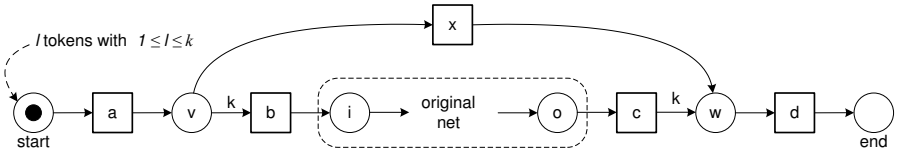


Fig. 12. The original WF-net is weak sound if and only if the constructed WF-net is up-to-k-sound.

N_k is up-to-k-sound and therefore the latter is also undecidable. Note that if less than k tokens are put into the source place, the original net N is not activated and the tokens bypass N via x. Therefore, k-soundness of the net in Figure 11 is the same as up-to-k-soundness of the net in Figure 12. □

Using Corollary 1 it can be shown that up-to-k-soundness is also undecidable for WF-nets with inhibitor arcs.

6.7 Generalized soundness

While weak-soundness and k-soundness are closely related to classical soundness, generalized soundness is quite different because it marks the source place with an arbitrary (i.e., non-predefined) number of tokens. Let us first consider the situation without reset and/or inhibitor arcs. Even in this simple setting it is not possible to use the coverability graph [62] to decide soundness. The reason is that the problem corresponds to inspecting infinitely many coverability graphs. Fortunately, in [36] it was shown that generalized soundness is decidable for WF-nets without reset and/or inhibitor arcs.

Theorem 2 (Decidability of generalized soundness [36]). *Generalized soundness is decidable for WF-nets without reset and/or inhibitor arcs.*

Proof. See [36]. □

Let us now consider a WF-net with inhibitor arcs and no reset arcs. It is well-known that the reachability problem is undecidable for Petri nets with inhibitor arcs [16]. This can be used to prove that generalized soundness is undecidable.

Proposition 3 (Undecidability of generalized soundness). *Generalized soundness is undecidable for WF-nets with inhibitor arcs.*

Proof. Let N be an arbitrary inhibitor WF-net. Let N' be the net obtained using the construction shown in Figure 13.

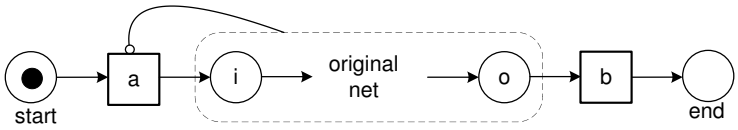


Fig. 13. The original WF-net is weak sound if and only if the constructed WF-net is generalized sound. Note that a short-hand notation is used; transition a has an inhibitor arc for each place in the original net.

For any k and any reachable state of $(N', [start^k])$, a blocks if there is still a token in the original net and will continue to block until b removes the last token from N.

Assume N is weak sound. It is easy to see that any token put into i, can always evolve to a state with a token in o and where the rest of the original net is empty and a continues to block. In this state b can fire and this can be repeated for all k tokens initially put in place *start*. Hence N' is generalized sound.

Assume N' is generalized sound. Hence the net N' is also weak sound which implies that N is weak sound.

Therefore, N' is generalized sound if and only if N is weak sound. Since weak soundness is undecidable for WF-nets with inhibitor arcs, generalized soundness is also undecidable. □

In Figure 13, the original WF-net N is activated only once, i.e., a blocks any new activations until b removes the last token from N. This is sufficient for proving undecidability. However, using Figure 14, we also explore the reverse situation. Let N be the original net in Figure 14 and let N' be the extended WF-net as shown in the same figure. N is generalized sound if and only if N' is weak sound. This construction provides additional insight in the relation between weak and generalized soundness.

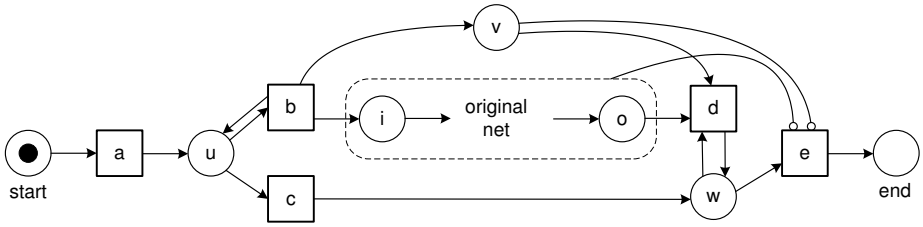


Fig. 14. Construction showing that generalized soundness can be expressed in terms of weak soundness for WF-nets with inhibitor arcs.

Generalized soundness is decidable for WF-nets without reset/inhibitor arcs and undecidable for WF-nets with inhibitor arcs. Therefore, the remaining question is: "Is generalized soundness decidable for WF-nets with reset arcs?". The answer to this question is unknown, i.e., it is still an open problem.

One may think that a construction similar to Figure 13 is possible for reset arcs. Such an attempt is made in Figure 15. Let N be the original net in Figure 15 and let N' be the extended WF-net as shown in the same figure. The goal would be to show that N is generalized sound if and only if N' is weak sound. However, this construction does not work because the N' produces a single token for place *end*, independent of the number of tokens initially put in *start*. The thing that is missing is a "counter" like place v in Figure 14. However, using reset arcs it is impossible to make such a construction. It is also not possible to use the construction of Theorem 1, because if multiple cases enter the construction, the original net fragment is able to reach states not reachable from M_I. It is not

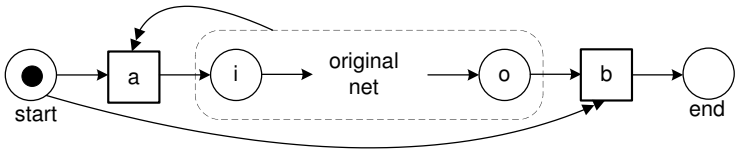


Fig. 15. Failed attempt to show that generalized soundness is undecidable for WF-nets with reset arcs.

possible to temporarily block the entry of additional cases, because the WF-net starts empty (i.e., no "mutex place" is possible) and no inhibitor arcs are allowed. Therefore, decidability of generalized soundness for WF-nets with reset arcs remains an open problem.

6.8 Relaxed soundness

Relaxed soundness differs fundamentally from notions such as classical, weak, and generalized soundness, because it allows for deadlocks, etc. as long as there is a "good execution" possible for each transition.

Theorem 3 (Undecidability of relaxed soundness). *Relaxed soundness is undecidable for WF-nets with reset arcs.*

Proof. Let (N, M_I) be an arbitrary marked Petri net with reset arcs and without inhibitor arcs. Without loss of generality we can assume that N is connected and that every transition has input and output places. Any net can be translated into a behaviorally equivalent net that has these properties.

To show that relaxed soundness is undecidable, we construct a new net $(N', [i])$ which embeds (N, M_I) such that N' is relaxed sound if and only if some marking M_X is reachable from (N, M_I). By doing so, we show that reachability in an arbitrary reset net can be analyzed through relaxed soundness, making relaxed soundness undecidable because reachability is undecidable for reset nets [22, 23].

Note that here we choose a different strategy than in Theorem 1 where soundness corresponds to the *non*-reachability of a given marking M_I. Here, we make a construction such that the relaxed soundness of N' corresponds to the reachability of M_I in (N, M_I).

Figure 16 shows the basic idea of the constructing N' from N. P is the set of places in N and T is the set of transitions in N. $I \subseteq P$ is the set of places marked in M_I and $X \subseteq P$ is the set of places marked in M_X. Although not shown in Figure 16, I and X may overlap. Let $T_{start} = \{t_{start} \mid t \in T\}$ and $T_{end} = \{t_{end} \mid t \in T\}$ be new transitions and let $S = \{s_t \mid t \in T\}$ be new places, i.e., for each $t \in T$ we add a place s_t and transitions t_{start} and t_{end}. Assume $(\{i, o, u, v, w\} \cup S) \cap P = \emptyset$ and $(\{a, b, c\} \cup T_{start} \cup T_{end}) \cap T = \emptyset$.

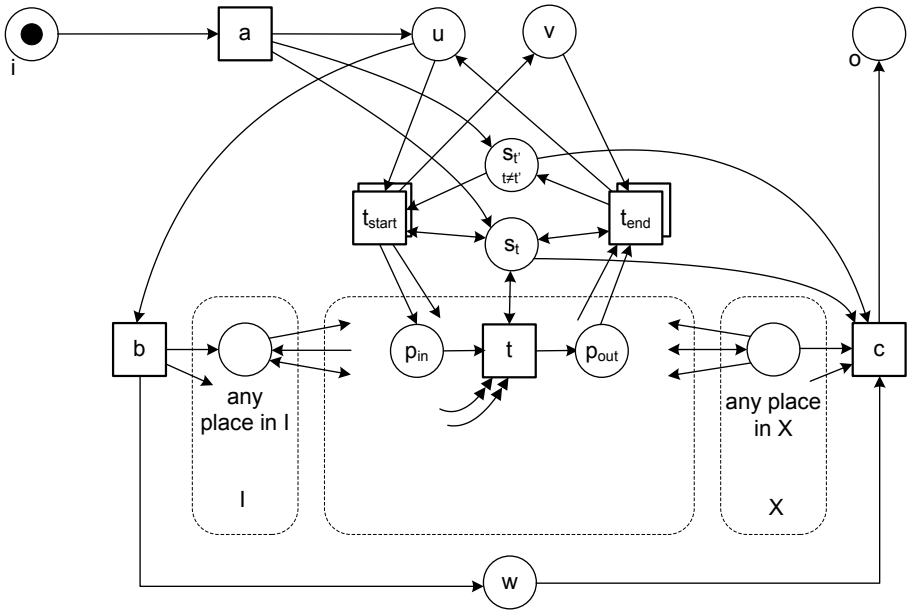


Fig. 16. Construction showing that reachability can be expressed in terms of relaxed soundness for WF-nets with reset arcs.

For any t: $\bullet t_{start} = [u] + S$, $t_{start}\bullet = (\bullet t) + [s_t, v]$, $\bullet t_{end} = (t\bullet) + [s_t, v]$, and $t_{end}\bullet = [u] + S$. As Figure 16 shows, transition b initializes the places in I, i.e., for $p \in I$: $W(b,p) = M_I(p)$. Similarly, transition c consumes all tokens from X if marking M_X is reached, i.e., for $p \in X$: $W(p,c) = M_X(p)$.

To better understand the structure of N' note that there are the following place invariants: $i+u+v+w+o$ and $k.i+\sum_{t\in T} s_t+(k-1).v+k.o$ where $k = |T|$. The first invariant indicates that there will always be one token in exactly one of the places i, u, v, w, and o. The second invariant shows that there is a token in i (weight k), or there is a token in o (weight k), or there are tokens in $S \cup \{v\}$. In the latter case, there may be one token in v with weight $k-1$ and one token in one of the places in S with weight 1. So the sum of these two tokens is also k. Note that t_{start} consumes k tokens with weight one from S, returns one token to place $s_t \in S$, and puts a token with weight $k-1$ in place v. Transition t_{end} consumes one token from place $s_t \in S$ and one token with weight $k-1$ for place v, and produces k tokens with weight one for S. It is easy to show that these are indeed invariants because the reset arcs only affect the places in P and not any of the newly added places.

Initially a fires thus marking u and all places in S. In $[u] + S$, any of the T_{start} transitions can fire. Say t_{start} fires. In the resulting state $((\bullet t) + [s_t, v])$, t is the only transition that can fire. Note that all other transitions in T are blocked because the corresponding places in $S \setminus \{s_t\}$ are not marked. After firing

t, t_{end} is the only transition that can fire. Note that reset arcs do not play a role here because transition t removes the tokens in $\bullet t$ and nothing more. Firing the sequence $\langle t_{start}, t, t_{end}\rangle$ results again in marking $[u]+S$. Hence this could be repeated for all $t \in T$, still resulting in marking $[u]+S$. In marking $[u]+S$ also b can fire resulting in marking $M_I + S + [w]$. Hence is it possible to move from marking $[i]$ to marking $M_I + S + [w]$ by firing $\sigma_b = \langle a, \ldots, t_{start}, t, t_{end}, \ldots, b\rangle$, i.e., $(N', [i])[\sigma_b\rangle(N', M_I + S + [w])$. Note that σ_b contains all transitions except c. After executing σ_b, the transitions in T can fire like in (N, M_I), i.e., not constrained by the added constructs, until c occurs. When c occurs all tokens in S are removed thus blocking all transitions in T. After firing c a token is put into o and no transition can fire anymore. After c fires the net is in a dead state that at least covers marking $[o]$.

Now we can show that N' is relaxed sound if and only if some marking M_X is reachable in (N, M_I):

- Assume marking M_X is reachable from (N, M_I). There exists a firing sequence σ_N such that $(N, M_I)[\sigma_N\rangle(N, M_X)$. This sequence is also enabled in the state after executing σ_b: $(N', M_I + S + [w])[\sigma_N\rangle(N', M_X + S + [w])$. Hence, $(N', [i])[\sigma_b\sigma_N c\rangle(N', [o])$ and it becomes clear that N' is indeed relaxed sound.
- Assume N' is relaxed sound. Hence there is a sequence σ: $(N', [i])[\sigma\rangle(N', [o])$. σ needs to have the following structure $\sigma_b = \langle a, \ldots, b, \ldots, c\rangle$ because in order to mark o, c must have been the last step and must have been preceded by b which in turn must have been preceded by a. Recall that $i + u + v + w + o$ is a place invariant illustrating the main control-flow in the net and the linear dependencies between a, b and c. It is also clear that a, b, and c can fire only once. Just before firing c the marking must have been precisely $M_X + S + [w]$ because c does not have any reset arcs. Just after firing b the marking must have been $M_I + S + [w]$. Hence, there exists a firing sequence σ_N such that $(N', M_I + S + [w])[\sigma_N\rangle(N', M_X + S + [w])$. Note that in σ_N only transitions of T can be present ($T_{start} \cup T_{end}$ are dead after removing the token from u). Hence, σ_N is also enabled in the original net, i.e., $(N, M_I)[\sigma_N\rangle(N, M_X)$. Therefore, M_X must be reachable in (N, M_I) thus completing the proof.

□

Combining Theorem 3 and Proposition 2 shows that relaxed soundness is also undecidable for WF-nets with inhibitor arcs. However, it is also possible to provide a direct proof using the same construction as shown in Figure 16. Note that the construction does not use any reset arcs and is therefore also applicable in this case. The same sequence $\sigma_b = \langle a, \ldots, t_{start}, t, t_{end}, \ldots, b\rangle$ used in the proof of Theorem 3 is still enabled if the transitions are assumed to be not trivially dead (i.e., an inhibitor arc connected to an input place). Moreover, after firing b the net is guaranteed to be in $M_I + S + [w]$. Hence the same arguments apply.

Relaxed soundness is decidable for WF-nets without reset/inhibitor arcs. In most cases, a direct inspection of the coverability graph will be sufficient to conclude this. Often one can also use a partially constructed reachability

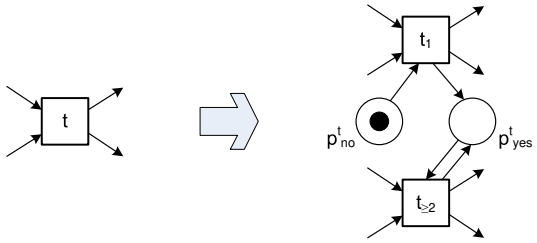


Fig. 17. Construction showing that relaxed soundness is decidable for WF-nets without reset/inhibitor arcs, i.e., t is involved in a "good run" in the original model (left) if and only if $[o, p^t_{yes}]$ is reachable in the adapted model (right).

graph to show relaxed soundness. If the net is unbounded, there may be cases where such simple inspections are inconclusive. A conclusive strategy to check for relaxed soundness would be to add two places (e.g., p^t_{yes} and p^t_{no}) for transition t that record whether t has been fired or not. Note that transition t needs to be duplicated into t_1 and $t_{\geq 2}$ as shown in Figure 17. Transition t_1 corresponds to the first execution of t while $t_{\geq 2}$ corresponds to later executions. A "good execution" in the original model leads from $[i]$ to $[o]$. Now the reachability of state $[o, p^t_{yes}]$ corresponds to a "good" execution sequence where t occurred at least once. This can be repeated for all transitions, and, since reachability is decidable for classical Petri nets, this implies that relaxed soundness is decidable for WF-nets without reset/inhibitor arcs.

6.9 Lazy soundness

Lazy soundness focuses on the marking of place o and does not require the net to be empty after putting a token in o. Nevertheless, we can use the construction of Theorem 1 to show that the property is undecidable for WF-nets with reset arcs

Theorem 4 (Undecidability of lazy soundness). *Lazy soundness is undecidable for WF-nets with reset arcs.*

Proof. To show that lazy soundness is undecidable, we again use the construction shown in Figure 10. A new net $(N', [i])$ is created which embeds (N, M_I) such that N' is lazy sound if and only if some marking M_X is NOT reachable from (N, M_I).

- Assume marking M_X is reachable from (N, M_I). This implies that from $(N', [i])$ the deadlock marking $[s, w]$ is reachable. Hence there is a scenario where place o cannot be marked. Hence N' is not lazy sound.
- Assume marking M_X is not reachable from (N, M_I) and M_X is also not coverable. Hence, it is always possible to terminate in state $[o]$ by firing one

of the "z transitions" because one of the places in P will remain marked because b cannot fire. This is the only reachable state marking o. Therefore, N' is indeed lazy sound.
– Assume marking M_X is not reachable from (N, M_I) but M_X is coverable. This implies that in the embedded subnet it is only possible to reach states that do not cover M_X or that are bigger than M_X. For states smaller than M_X we have shown that lazy soundness is not jeopardized. For states bigger than M_X, b can fire. However, if b fires, tokens remain in P and b cannot fire anymore. Hence, at least one "z transition" is enabled at any time and, as a result, it is always possible to terminate in state $[o]$. This implies that N' is indeed lazy sound.

□

As a result, lazy soundness is also undecidable for WF-nets with inhibitor arcs. The property is, however, decidable for WF-nets without reset and/or inhibitor arcs, e.g., by inspecting the coverability graph.

6.10 Easy soundness

The last soundness notion we consider is easy soundness. Recall that this notion simply checks whether there is an execution path from $[i]$ to $[o]$.

Theorem 5 (Undecidability of easy soundness). *Easy soundness is undecidable for WF-nets with reset arcs.*

Proof. Let (N, M_I) be an arbitrarily marked Petri net with reset arcs but no inhibitors. Without loss of generality we again assume that N is connected and that every transition has input and output places.

To show that easy soundness is undecidable, we construct a new net $(N', [i])$ which embeds (N, M_I) such that N' is easy sound if and only if some marking M_X is reachable from (N, M_I). By doing so, we show that reachability in an arbitrary reset net can be analyzed through soundness, making easy soundness also undecidable.

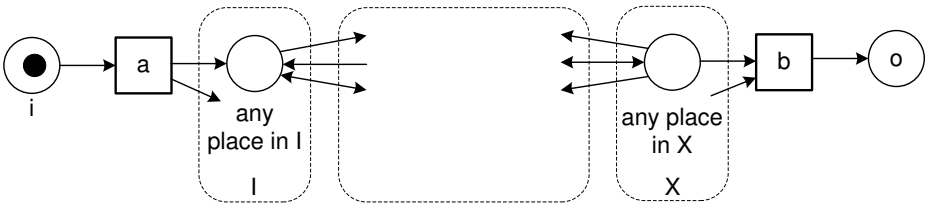


Fig. 18. Construction showing that easy soundness is undecidable for WF-nets with reset arcs.

Figure 18 shows N and N' using the notation used before.

– Assume marking M_X is reachable from (N, M_I). This implies that M_X is also reachable in $(N', [i])$ and that b can fire without leaving any tokens behind in the set of places of the original net. Hence, firing b results in $[o]$ and clearly the net is easy sound.
– Assume N' is easy sound, i.e., there is a firing sequence leading from $[i]$ to $[o]$. Hence there was a moment when b fired. If at this point in time the marking was not exactly M_X, then tokens are left behind in the original net. Since all transitions have input and output places, the remaining tokens cannot be removed completely and hence all subsequent markings are larger than $[o]$. Hence, just before b fired, the marking was exactly M_X showing that M_X is indeed reachable from (N, M_I).

□

Easy soundness is therefore also undecidable for WF-nets with inhibitors. The property is decidable for WF-nets without reset/inhibitor arcs because reachability is decidable for classical Petri nets.

6.11 Overview

Table 1 summarizes the results presented in this section. As shown, all eight soundness properties are decidable for WF-nets without reset and/or inhibitor arcs. For WF-nets with reset arcs the decidability of generalized soundness is still unknown. For all other cases, soundness is undecidable. As Table 1 indicates, it is difficult to precisely analyze models in more advanced languages having features which correspond to reset or inhibitor arcs. Nevertheless, there are pragmatic approaches that allow for the discovery of design errors even if such features are present. This will be explained further in the next section.

Table 1. Decidability of soundness (U=Undecidable, D=Decidable, and ?=unknown).

	(classical) soundness	weak soundness	k-soundness	up-to-k soundness	generalized soundness	relaxed soundness	lazy soundness	easy soundness
no reset/inhibitor arcs	D	D	D	D	D	D	D	D
just reset arcs	U	U	U	U	?	U	U	U
just inhibitor arcs	U	U	U	U	U	U	U	U
reset and inhibitor arcs	U	U	U	U	U	U	U	U

7 Analysis

In this section, we focus more on the pragmatic side of workflow verification. As illustrated by Table 1 one can roughly say that for WF-nets without reset and/or inhibitor arcs all soundness properties are decidable while for extended WF-nets (e.g., nets with reset and/or inhibitor arcs) these notions are undecidable. Note that most workflow and process modeling languages (e.g., BPMN, EPCs, Staffware, BPEL, FileNet, etc.) have a control-flow language where the basic elements correspond to WF-nets without reset and/or inhibitor arcs, while the more advanced constructs require reset or inhibitor arcs. Therefore, one can argue that, as a rule-of-thumb, for simple models (independent of the language used) any form of soundness is decidable while for models using notions such as priorities, cancellation, etc. no form of soundness is decidable. However, even if more advanced constructs are used, workflow verification is still possible! Note that even if soundness is undecidable for a particular class of WF-nets, for many representatives of such a class, it may still be possible to conclude soundness or non-soundness. There may be rules of the form "If WF-net N has property X, then N is sound" or "If WF-net N has property Y, then N is not sound". As shown in [57, 59, 58, 60] it is possible to find many errors using such an approach. In [58, 60] it was shown that at least 5.6 percent of the process models in SAP's reference model are not sound. However, this 5.6 percent is merely a lower bound. In [57, 59] an even larger set of more than 2000 process models from practice was analyzed. It could be shown that at least 10 percent of these models is not sound. These examples show that *even if soundness is undecidable, errors can be discovered*. Similarly, for many models it is still possible to guarantee soundness even if the general verification problem is undecidable.

As for many specific process models it is possible to make useful conclusions even if the general question is undecidable, we advocate a more down-to-earth approach. In this section, we discuss three analysis approaches in relation to the 4*8=32 verification problems portrayed in Table 1.

7.1 Coverability graph

The construction of a coverability graph is one of the standard approaches for analyzing classical Petri nets [62, 65]. If the state-space is finite, the coverability graph coincides with the reachability graph where nodes correspond to reachable markings and arcs correspond to state transitions. If the state-space is infinite, the coverability graph contains so-called ω markings indicating that the number of tokens on a particular place may grow unbounded. Figure 19 shows a WF-net (please ignore the dashed arcs for the moment) and the corresponding coverability graph is shown in Figure 20.

Based on the coverability graph one can see that the WF-net without the dashed arcs, i.e., net N, is not sound (classical soundness). Node $p5 + \omega.p3$ in Figure 20 indicates that it is possible to reach a state with one token in $p5$ and

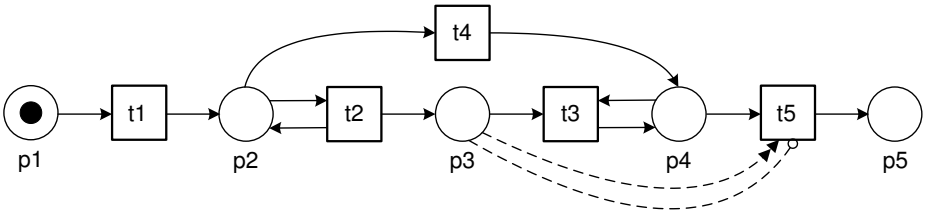


Fig. 19. N is the WF-net without the reset/inhibitor arc, N_R is the net with the reset arc, N_I is the net with the inhibitor arc, and N_{RI} is the net with both the reset arc and inhibitor arc.

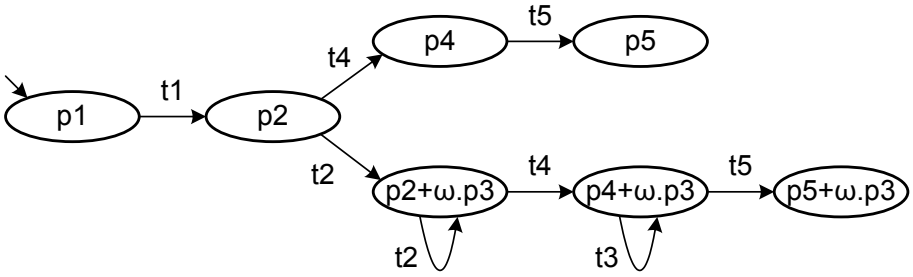


Fig. 20. The coverability graph of N.

an arbitrarily large number of tokens in $p3$.[9] This proves that N is not sound (no proper completion). Moreover, it can be used to show that N is also not weak sound. For any $k \geq 2$ the coverability graph can be constructed and this will show that the net is also not k-sound. Therefore, it is possible to show that N is also not up-to-k sound or generalized sound. However, using the coverability graph in Figure 20 it can be shown that N is lazy sound and easy sound. Using the construction illustrated in Figure 17, a set of adapted coverability graphs can be used to prove relaxed soundness. In general, the coverability graph (or a set of coverability graphs) can be used to decide on any type of soundness except for generalized soundness. For generalized soundness one can use the approach described in [36].

Let us consider again the WF-net shown in Figure 19. N_R is the net with the dashed reset arc and without the dashed inhibitor arc. N_R is classical sound, weak sound, k-sound, up-to-k sound, generalized sound, relaxed sound, lazy sound, and easy sound. N_I is the net with the inhibitor arc and not the reset arc and N_{RI} is the net with both arcs. N_I and N_{RI} also satisfy all eight notions of soundness.

In the presence of reset or inhibitor arcs, the coverability graph cannot be used to come to a conclusive answer. If the state space is finite, the reachability graph can be constructed and seven notions of soundness can be checked (all except generalized soundness). Note however that boundedness is undecidable for nets with reset or inhibitor arcs [23]. Nevertheless, the coverability graph of a WF-net where the reset and inhibitor arcs are ignored, provides a kind of "upperbound" for the model with reset and/or inhibitor arcs. Let N_X be a WF-net such that after moving the reset and inhibitor arcs, N_X coincides with N in Figure 19. Based on the coverability graph in Figure 20, we can conclude that for such an N_X there can never be two tokens in $p5$ or a token in $p2$ and $p4$ at the same time. This illustrates that the coverability graph can be seen as an over-approximation of the true behavior.

7.2 Invariants

As shown in [3, 74] structural techniques can be used to analyze WF-nets without reset and/or inhibitor arcs. In this setting, the absence of certain invariants points towards errors as explained below.

A *place invariant* is a weighted sum over the places that is invariant under each possible transition firing. In Figure 19 the sum of tokens on the places $p1$, $p2$, $p4$, and $p5$ is constant independent of the initial marking, i.e., it is a structural property. The absence of particular place invariants hints at problems. For example, in WF-net N shown in Figure 19 there is no place invariant adding a positive weight to all places. This typically suggests some structural anomaly. In this case, there is a positive invariant involving all places except $p3$ and there is no such invariant involving $p3$. This is caused by the fact that $p3$ is unbounded

[9] To be more precise: for any $k \in \mathbb{N}$ there exists an $l \geq k$ such that state $[p5, p3^l]$ is reachable.

and this is indeed the primary reason why N is not classical/weak sound. Hence, the lack of such an invariant is a useful diagnostic.

A *transition invariant* assigns a weight to all transitions such that if each transition is able to fire the number of times indicated by the weight, the system is back in the initial state. Note that it is not guaranteed that a transition invariant is realizable, i.e., it is a structural property independent of the initial marking and there may be too few tokens to allow each of the transitions fire the designated number of times. When applying transition invariants one should consider the so-called short-circuited net mentioned earlier. The short-circuited net is the Petri net obtained by connecting o to i via a new transition t^* thus making the net cyclic. When considering classical soundness or relaxed soundness, there should be a semi-positive transition invariant for each transition, i.e., for each transition t it should be possible to find an invariant that assigns a positive weight to both t and t^*. If it is not possible to find such an invariant for t, the transition cannot contribute to any execution sequence leading from $[i]$ to $[o]$.

The above shows that *using invariants one can generate useful diagnostics* for WF-nets without reset and/or inhibitor arcs. It is easy to see that invariants are *less useful for nets with reset arcs* because these arcs destroy the nice linear algebraic properties that follow from the marking equation [62, 65]. However, *all the properties still hold for WF-nets with inhibitor arcs*. This is shown in detail in [73]. Consider N_I in Figure 19 (i.e., the WF-net with the inhibitor arc). If N_I is classical sound or relaxed sound, then for each t here has to be an invariant that assigns a positive weight to t and the short-circuiting transition t^*. This is indeed the case. In fact, the transition invariant that assigns weight 1 to all transitions including t^* is such an invariant. If such invariant(s) would not exist, the WF-net with the inhibitor arc could not be classical/relaxed sound. Also note that inhibitor arcs leave place invariants intact, i.e., the behavioral properties of place invariants obtained by ignoring inhibitor arcs still hold when inhibitor arcs are added.

In a practical setting, invariants can be used to discover lots of errors. For example, in [58, 60] it is shown that many errors in the SAP reference model can be discovered using transition invariants.

7.3 Reduction rules

The construction of a coverability graph can be considered as a "brute force" approach while the use of structural techniques such as invariants aim at a more efficient analysis. However, in many cases the "brute force" takes too long and structural techniques give inconclusive answers. Reductions rules provide a different approach and can be used in combination with other techniques. The goal of a reduction rule is to make the net smaller without changing specific properties such as soundness. There are basically two reasons for using reduction rules in the context of WF-nets. First of all, by making the WF-net smaller without changing its soundness properties, it becomes easier to apply other methods such as the construction of the coverability graph. Second, using reduction rules it is

possible to provide better diagnostics, i.e., by reducing the trivially correct parts of a WF-net, the focus can be on the erroneous or suspicious constructs that remain.

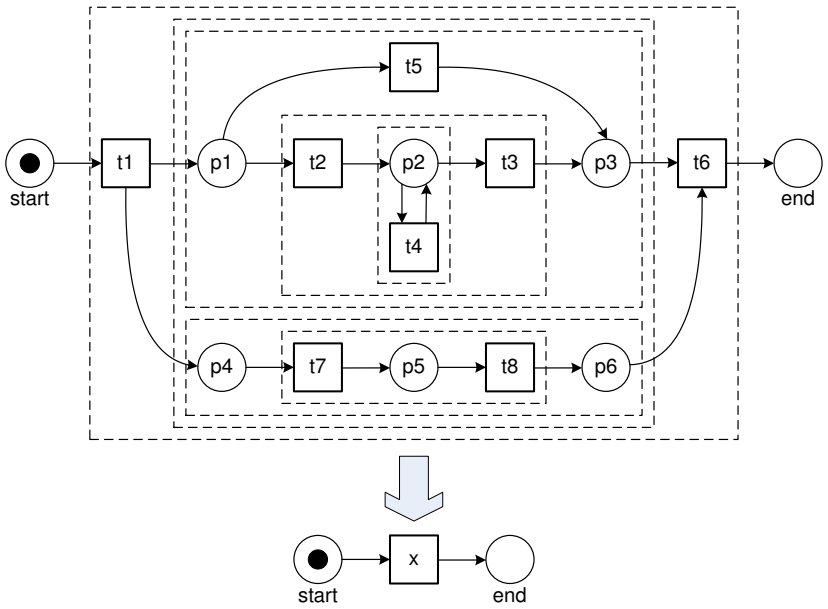


Fig. 21. A sound WF-net that can be reduced to the trivially correct WF-net.

To illustrate the use of reduction rules, we consider the sound WF-net shown in Figure 21 (top one). Note that this net satisfies all eight forms of soundness. We can apply the liveness and boundedness preserving reduction rules described in [12, 13, 62] and by doing so we obtain the net shown at the bottom of Figure 21. The dashed lines illustrate the scopes of the various applications of the reduction rules. The reduced model is trivially sound, so the original model was also sound. We can apply the liveness and boundedness preserving reduction rules of [12, 13, 62] to get this result, because a WF-net is classical sound if and only if the corresponding short-circuited net is live and bounded [1, 2].

Figure 22 shows how reduction rules can be used to show that a WF-net is not sound and highlights the problematic parts. By applying the liveness and boundedness preserving reduction rules, we can reduce the top net into the small net at the bottom. The reduced net clearly shows the problem, i.e., a lack of synchronization. Note that the short-circuited counterparts of both WF-nets are unbounded and hence these WF-nets are not sound. Note that all the classical liveness and boundedness preserving reduction rules [12, 13, 62] that do not depend on the initial marking also preserve the other notions of soundness.

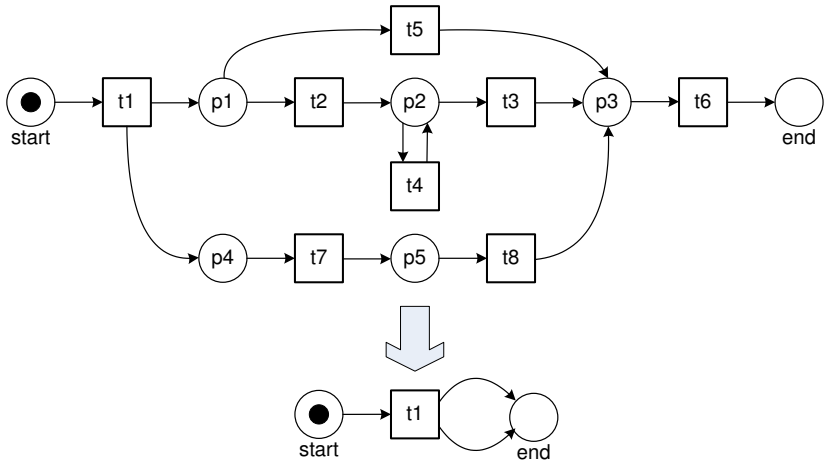


Fig. 22. A WF-net that is not sound and that cannot be reduced completely.

However, these rules do not necessarily apply to Petri nets with reset and/or inhibitor arcs.

In [75, 81] the liveness and boundedness preserving reduction rules are extended to reset and/or inhibitor arcs, i.e., new rules are given and existing rules are modified. This set of rules can be applied to WF-nets such as the one shown in Figure 23. The top net is classical sound, but is a bit difficult to interpret. (Note that the net is also weak and relaxed sound, but not 2-sound.) Using the rules presented in [75, 81] the net can be reduced without changing things with respect to weak/classical soundness. In the resulting WF-net only half of the transitions remain.

A detailed discussion on the application of reduction rules to soundness analysis is beyond the scope of this paper. However, we informally show one of the reduction rules that preserves liveness and boundedness for nets with reset and/or inhibitor arcs. Using the *Fusion of Series Places* (FSP) rule shown in Figure 24, we can reduce two places and one transition to one place. Thus, this rule effectively removes a transition and a place. The conditions are sketched informally in Figure 24. Tokens which reside in the first place p can be considered to be "ghost tokens" for the second place q. Since there is no transition having p as input place other than t, tokens in p can/will always end up in q. If some transition needs to consume a ghost token in p, the intermediate transition t should fire first, replacing the ghost token in p by a tangible one in q. Since t should not be blocked and have other side-effects, t should not have any reset or inhibitor arcs. There can be reset and/or inhibitor arcs connected to p and q. However, these two places should be "identical" in terms of such arcs. Under such circumstances, p and q can be merged into r and t is no longer needed. The net before applying the FSP rule is live if and only if the net after applying the

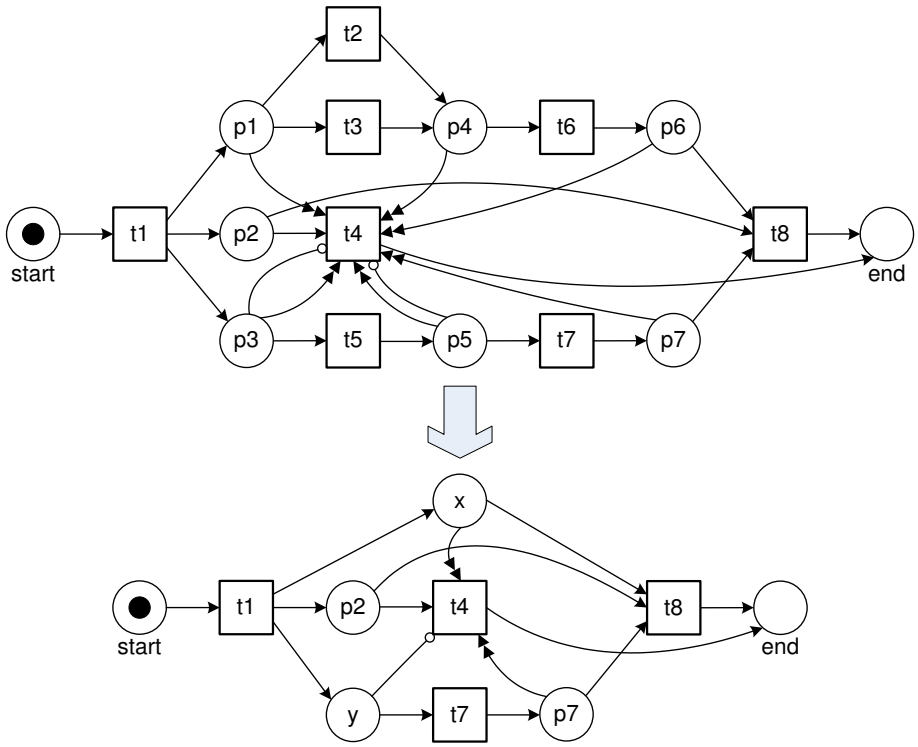


Fig. 23. Partial reduction of a WF-net with reset and inhibitor arcs.

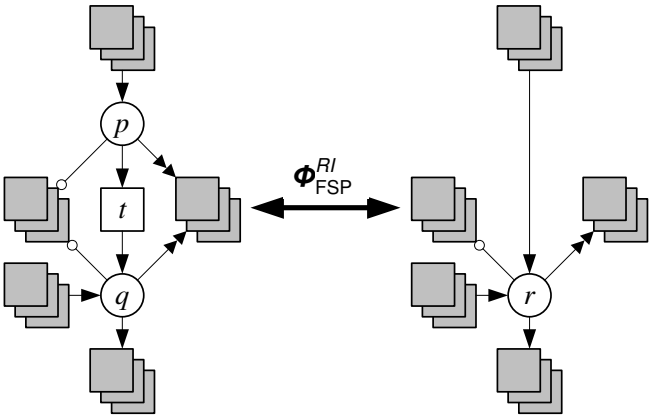


Fig. 24. One of the liveness and boundedness preserving reduction rules presented in [75]: Fusion of Series of Places (FSP).

FSP rule is live. Moreover, the net before applying the FSP rule is bounded if and only if the net after applying the FSP rule is bounded. This is proven in [75]. Note that the FSP rule was applied repeatedly in figures 21, 22, and 23. For WF-nets without reset and inhibitor arcs, soundness corresponds to liveness and boundedness. This is not the case for WF-nets with reset and inhibitor arcs. However, it is easy to prove that the the FSP rule also preserves soundness for arbitrary WF-nets with reset and inhibitor arcs.

The above examples illustrate that reduction rules can be used to simplify the problem and provide diagnostics. For particular languages variants of the above rules can be developed. See [21] for a small set of reduction rules for EPCs. In [57, 59] a more extensive set of reduction rules for EPCs is given. Using these rules in combination with state space analysis, many errors were found in a sample of more than 2000 non-trivial EPCs from industry [57, 59]. Moreover, the reduction rules could also be used to highlight the errors.

In this section, we discussed three different approaches to tackle workflow verification problems. The goal was not to present a particular analysis technique but to provide an overview. This overview shows that for WF-nets without reset and/or inhibitor arcs, standard techniques such as the coverability graph and reduction rules can be used to decide soundness. Moreover, even though reset and/or inhibitor arcs are present and soundness is in principle undecidable, often it is still possible to apply these techniques and obtain valuable answers. Empirical studies show that in such cases many errors can be discovered even if soundness is undecidable in the general case.

8 Conclusion

Over the last decade many papers on workflow verification have appeared. Some of these papers use Petri nets as a representation language. Other papers use a wide variety of similar graph-based languages (e.g., EPCs, BPMN, or simple AND-XOR graphs). However, independent of the representation used, there is always the basic notion of the creation of a process instance (case) and the successful completion of it. This naturally leads to various notions of *soundness*, i.e., reasoning about the correctness of a workflow model without any domain knowledge. Typical ingredients are absence of deadlocks and livelocks, proper termination, non-dead tasks, etc. Different soundness notions have been presented in the literature. Therefore, this paper provides a rigorous analysis of the different notions of soundness. Decidability of the various soundness notions has not been systematically investigated before. In fact, *this is the first paper to present decidability results for soundness in the presence of reset and/or inhibitor arcs.*

In order to investigate the various soundness notions, we use the so-called *extended WF-nets* as a starting point. These nets support the basic routing constructs but also allow for more advanced patterns through the so-called reset and inhibitor arcs. Notions such as cancellation, priority, etc. can be modeled

using such arcs. As shown in Table 1, 4*8=32 verification problems have been investigated for four classes of WF-nets and eight notions of soundness. For WF-nets without reset and/or inhibitor arcs all eight notions of soundness are decidable. However, as shown in this paper, most (if not all) notions of soundness become undecidable when reset and/or inhibitor arcs are used. Only for the combination of generalized soundness and WF-nets with reset arcs, did we not find a conclusive answer. This remains an open problem and a topic for future research.

As explained in Section 7, undecidability does not make things hopeless. Many errors can be discovered using techniques such as invariants and reduction rules. Applying such techniques to real-life models typically results in the discovery of many errors. This has been demonstrated in the few empirical studies that are available [58, 60, 57, 59, 71]. Currently, we are planning more empirical studies on workflow verification. We consider this more important than inventing new (toy) workflow notations and their corresponding soundness properties.

References

1. W.M.P. van der Aalst. Verification of Workflow Nets. In P. Azéma and G. Balbo, editors, *Application and Theory of Petri Nets 1997*, volume 1248 of *Lecture Notes in Computer Science*, pages 407–426. Springer-Verlag, Berlin, 1997.
2. W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.
3. W.M.P. van der Aalst. Workflow Verification: Finding Control-Flow Errors using Petri-net-based Techniques. In W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*, pages 161–183. Springer-Verlag, Berlin, 2000.
4. W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA, 2004.
5. W.M.P. van der Aalst, A. Hirnschall, and H.M.W. Verbeek. An Alternative Way to Analyze Workflow Graphs. In A. Banks-Pidduck, J. Mylopoulos, C.C. Woo, and M.T. Ozsu, editors, *Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE'02)*, volume 2348 of *Lecture Notes in Computer Science*, pages 535–552. Springer-Verlag, Berlin, 2002.
6. W.M.P. van der Aalst and A.H.M. ter Hofstede. Verification of Workflow Task Structures: A Petri-net-based Approach. *Information Systems*, 25(1):43–69, 2000.
7. W.M.P. van der Aalst and A.H.M. ter Hofstede. YAWL: Yet Another Workflow Language. *Information Systems*, 30(4):245–275, 2005.
8. W.M.P. van der Aalst, N. Lohmann, P. Massuthe, C. Stahl, and K. Wolf. From Public Views to Private Views: Correctness-by-Design for Services. In M. Dumas and H. Heckel, editors, *Informal Proceedings of the 4th International Workshop on Web Services and Formal Methods (WS-FM 2007)*, pages 119–134. QUT, Brisbane, Australia, 2007.
9. K. Barkaoui and L. Petrucci. Structural Analysis of Workflow Nets with Shared Resources. In W.M.P. van der Aalst, G. De Michelis, and C.A. Ellis, editors, *Proceedings of Workflow Management: Net-based Concepts, Models, Techniques and Tools (WFM'98)*, volume 98/7 of *Computing Science Reports*, pages 82–95, Lisbon, Portugal, 1998. Eindhoven University of Technology, Eindhoven.

10. A. Basu and R.W. Blanning. A Formal Approach to Workflow Analysis. *Information Systems Research*, 11(1):17–36, 2000.
11. A. Basu and A. Kumar. Research Commentary: Workflow Management Issues in e-Business . *Information Systems Research*, 13(1):1–14, 2002.
12. G. Berthelot. Checking Properties of Nets Using Transformations. In G. Rozenberg, editor, *Advances in Petri Nets 1985*, volume 222 of *Lecture Notes in Computer Science*, pages 19–40. Springer-Verlag, Berlin, 1986.
13. G. Berthelot. Transformations and Decompositions of Nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Advances in Petri Nets 1986 Part I: Petri Nets, central models and their properties*, volume 254 of *Lecture Notes in Computer Science*, pages 360–376. Springer-Verlag, Berlin, 1987.
14. H.H. Bi and J.L. Zhao. Applying Propositional Logic to Workflow Verification. *Information Technology and Management*, 5(3-4):293–318, 2004.
15. Y. Choi and J. Zhao. Decomposition-based Verification of Cyclic workflows. In D.A. Peled and Y-K. Tsay, editors, *Proceedings of Automated Technology for Verification and Analysis (ATVA 2005)*, volume 3707 of *Lecture Notes in Computer Science*, pages 84–98, Taipei, Taiwan, 2005. Springer-Verlag.
16. P. Chrzastowski-Wachtel. Testing Undecidability of the Reachability in Petri Nets with the Help of 10th Hilbert Problem. In S. Donatelli and J. Kleijn, editors, *Application and Theory of Petri Nets 1999*, volume 1639 of *Lecture Notes in Computer Science*, pages 268–281. Springer-Verlag, Berlin, 1999.
17. J. Dehnert. *A Methodology for Workflow Modeling: From Business Process Modeling Towards Sound Workflow Specification.* PhD thesis, TU Berlin, Berlin, Germany, 2003.
18. J. Dehnert and W.M.P. van der Aalst. Bridging the Gap Between Business Models and Workflow Specifications. *International Journal of Cooperative Information Systems*, 13(3):289–332, 2004.
19. J. Dehnert and P. Rittgen. Relaxed Soundness of Business Processes. In K.R. Dittrich, A. Geppert, and M.C. Norrie, editors, *Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01)*, volume 2068 of *Lecture Notes in Computer Science*, pages 157–170. Springer-Verlag, Berlin, 2001.
20. J. Desel and J. Esparza. *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, Cambridge, UK, 1995.
21. B.F. van Dongen, W.M.P. van der Aalst, and H.M.W. Verbeek. Verification of EPCs: Using Reduction Rules and Petri Nets. In O. Pastor and J. Falcão e Cunha, editors, *Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE'05)*, volume 3520 of *Lecture Notes in Computer Science*, pages 372–386. Springer-Verlag, Berlin, 2005.
22. C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset Nets Between Decidability and Undecidability. In K. Larsen, S. Skyum, and G. Winskel, editors, *Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, volume 1443 of *Lecture Notes in Computer Science*, pages 103–115, Aalborg, Denmark, July 1998. Springer-Verlag.
23. C. Dufourd, P. Jančar, and Ph. Schnoebelen. Boundedness of Reset P/T Nets. In J. Wiedermann, P. van Emde Boas, and M. Nielsen, editors, *Lectures on Concurrency and Petri Nets*, volume 1644 of *Lecture Notes in Computer Science*, pages 301–310, Prague, Czech Republic, July 1999. Springer-Verlag.

24. M. Dumas, W.M.P. van der Aalst, and A.H.M. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software through Process Technology*. Wiley & Sons, 2005.
25. J. Esparza. Decidability and Complexity of Petri Net Problems: An Introduction. In W. Reisig and G. Rozenberg, editors, *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*, pages 374–428. Springer-Verlag, Berlin, 1998.
26. J. Esparza. Reachability in Live and Safe Free-Choice Petri Nets is NP-Complete. *Theoretical Computer Science*, 198(1-2):211–224, 1998.
27. J. Esparza and M. Nielsen. Decidability Issues for Petri Nets: A Survey. *Journal of Information Processing and Cybernetics*, 30:143–160, 1994.
28. A. Finkel. The minimal coverability graph for Petri nets. In G. Rozenberg, editor, *Advances in Petri Nets 1993*, volume 674 of *Lecture Notes in Computer Science*, pages 210–243. Springer-Verlag, Berlin, 1993.
29. A. Finkel and Ph. Schnoebelen. Well-structured Transition Systems everywhere! *Theoretical Computer Science*, 256(1–2):63–92, April 2001.
30. X. Fu, T. Bultan, and J. Su. Formal Verification of e-Services and Workflows. In C. Bussler, R. Hull, S. McIlraith, M. Orlowska, B. Pernici, and J. Yang, editors, *Web Services, E-Business, and the Semantic Web, CAiSE 2002 International Workshop (WES 2002)*, volume 2512 of *Lecture Notes in Computer Science*, pages 188–202. Springer-Verlag, Berlin, 2002.
31. X. Fu, T. Bultan, and J. Su. Analysis of Interacting BPEL Web Services. In *International World Wide Web Conference: Proceedings of the 13th international conference on World Wide Web*, pages 621–630, New York, NY, USA, 2004. ACM Press.
32. D. Georgakopoulos, M. Hornick, and A. Sheth. An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3:119–153, 1995.
33. R.J. van Glabbeek and W.P. Weijland. Branching Time and Abstraction in Bisimulation Semantics. *Journal of the ACM*, 43(3):555–600, 1996.
34. K.M. van Hee, A. Serebrenik, N. Sidorova, and M. Voorhoeve. Soundness of Resource-Constrained Workflow Nets. In G. Ciardo and P. Darondeau, editors, *Applications and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 250–267. Springer-Verlag, Berlin, 2005.
35. K.M. van Hee, N. Sidorova, and M. Voorhoeve. Soundness and Separability of Workflow Nets in the Stepwise Refinement Approach. In W.M.P. van der Aalst and E. Best, editors, *Application and Theory of Petri Nets 2003*, volume 2679 of *Lecture Notes in Computer Science*, pages 335–354. Springer-Verlag, Berlin, 2003.
36. K.M. van Hee, N. Sidorova, and M. Voorhoeve. Generalised Soundness of Workflow Nets Is Decidable. In J. Cortadella and W. Reisig, editors, *Application and Theory of Petri Nets 2004*, volume 3099 of *Lecture Notes in Computer Science*, pages 197–215. Springer-Verlag, Berlin, 2004.
37. S. Hinz, K. Schmidt, and C. Stahl. Transforming BPEL to Petri Nets. In W.M.P. van der Aalst, A.H.M. ter Hofstede, and M. Weske, editors, *International Conference on Business Process Management (BPM 2005)*, volume 2678 of *Lecture Notes in Computer Science*, pages 220–235. Springer-Verlag, Berlin, 2005.
38. S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation*. International Thomson Computer Press, London, UK, 1996.

39. K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 1.* EATCS monographs on Theoretical Computer Science. Springer-Verlag, Berlin, 1997.
40. C. Karamanolis, D. Giannakopoulou, J. Magee, and S.M. Wheater. Model Checking of Workflow Schemas. In *Proceedings of the Fourth International Enterprise Distributed Object Computing Conference (EDOC'00)*, pages 170–181, Los Alamitos, CA, USA, 2000. IEEE Computer Society.
41. R.M. Karp and R.E. Miller. Parallel Program Schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969.
42. G. Keller, M. Nüttgens, and A.W. Scheer. Semantische Processmodellierung auf der Grundlage Ereignisgesteuerter Processketten (EPK). Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89 (in German), University of Saarland, Saarbrücken, 1992.
43. E. Kindler. On the Semantics of EPCs: A Framework for Resolving the Vicious Circle. *Data and Knowledge Engineering*, 56(1):23–40, 2006.
44. E. Kindler and W.M.P. van der Aalst. Liveness, Fairness, and Recurrence. *Information Processing Letters*, 70(6):269–274, June 1999.
45. E. Kindler, A. Martens, and W. Reisig. Inter-Operability of Workflow Applications: Local Criteria for Global Soundness. In W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*, pages 235–253. Springer-Verlag, Berlin, 2000.
46. P. Lawrence, editor. *Workflow Handbook 1997, Workflow Management Coalition.* John Wiley and Sons, New York, 1997.
47. F. Leymann and D. Roller. *Production Workflow: Concepts and Techniques.* Prentice-Hall PTR, Upper Saddle River, New Jersey, USA, 1999.
48. J. Li, Y. Fan, and M. Zhou. Performance Modeling and Analysis of Workflow. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 34(2):229–242, 2004.
49. H. Lin, Z. Zhao, H. Li, and Z. Chen. A Novel Graph Reduction Algorithm to Identify Structural Conflicts. In *Proceedings of the Thirty-Fourth Annual Hawaii International Conference on System Science (HICSS-35)*. IEEE Computer Society Press, 2002.
50. N. Lohmann, P. Massuthe, C. Stahl, and D. Weinberg. Analyzing Interacting BPEL Processes. In S. Dustdar, J.L. Faideiro, and A. Sheth, editors, *International Conference on Business Process Management (BPM 2006)*, volume 4102 of *Lecture Notes in Computer Science*, pages 17–32. Springer-Verlag, Berlin, 2006.
51. D.C. Marinescu. *Internet-Based Workflow Management: Towards a Semantic Web*, volume 40 of *Wiley Series on Parallel and Distributed Computing.* Wiley-Interscience, New York, 2002.
52. A. Martens. On Compatibility of Web Services. *Petri Net Newsletter*, 65:12–20, 2003.
53. A. Martens. Analyzing Web Service Based Business Processes. In M. Cerioli, editor, *Proceedings of the 8th International Conference on Fundamental Approaches to Software Engineering (FASE 2005)*, volume 3442 of *Lecture Notes in Computer Science*, pages 19–33. Springer-Verlag, Berlin, 2005.
54. A. Martens. Consistency between executable and abstract processes. In *Proceedings of International IEEE Conference on e-Technology, e-Commerce, and e-Services (EEE'05)*, pages 60–67. IEEE Computer Society Press, 2005.
55. P. Massuthe, W. Reisig, and K. Schmidt. An Operating Guideline Approach to the SOA. In *Proceedings of the 2nd South-East European Workshop on Formal Methods 2005 (SEEFM05)*, Ohrid, Republic of Macedonia, 2005.

56. P. Massuthe, W. Reisig, and K. Schmidt. An Operating Guideline Approach to the SOA. *Annals of Mathematics, Computing & Teleinformatics*, 1(3):35–43, 2005.
57. J. Mendling. *Detection and Prediction of Errors in EPC Business Process Models*. PhD thesis, Vienna University of Economics and Business Administration, Vienna, Austria, 2007.
58. J. Mendling, M. Moser, G. Neumann, H.M.W. Verbeek, B.F. van Dongen, and W.M.P. van der Aalst. Faulty EPCs in the SAP Reference Model. In S. Dustdar, J.L. Faideiro, and A. Sheth, editors, *International Conference on Business Process Management (BPM 2006)*, volume 4102 of *Lecture Notes in Computer Science*, pages 451–457. Springer-Verlag, Berlin, 2006.
59. J. Mendling, G. Neumann, and W.M.P. van der Aalst. Understanding the Occurrence of Errors in Process Models Based on Metrics. In F. Curbera, F. Leymann, and M. Weske, editors, *Proceedings of the OTM Conference on Cooperative information Systems (CoopIS 2007)*, volume 4803 of *Lecture Notes in Computer Science*, pages 113–130. Springer-Verlag, Berlin, 2007.
60. J. Mendling, H.M.W. Verbeek, B.F. van Dongen, W.M.P. van der Aalst, and G. Neumann. Detection and Prediction of Errors in EPCs of the SAP Reference Model. *Data and Knowledge Engineering*, 64(1):312–329, 2008.
61. M. zur Muehlen. *Workflow-based Process Controlling: Foundation, Design and Application of workflow-driven Process Information Systems*. Logos, Berlin, 2004.
62. T. Murata. Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.
63. F. Puhlmann and M. Weske. Interaction Soundness for Service Orchestrations. In A. Dan and W. Lamersdorf, editors, *Proceedings of Service-Oriented Computing (ICSOC 2006)*, volume 4294 of *Lecture Notes in Computer Science*, pages 302–313. Springer-Verlag, Berlin, 2006.
64. F. Puhlmann and M. Weske. Investigations on Soundness Regarding Lazy Activities. In S. Dustdar, J.L. Faideiro, and A. Sheth, editors, *International Conference on Business Process Management (BPM 2006)*, volume 4102 of *Lecture Notes in Computer Science*, pages 145–160. Springer-Verlag, Berlin, 2006.
65. W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998.
66. W. Sadiq and M.E. Orlowska. On Correctness Issues in Conceptual Modeling of Workflows. In *Proceedings of the 5th European Conference on Information Systems (ECIS '97)*, pages 19–21, Cork, Ireland, 1997.
67. W. Sadiq and M.E. Orlowska. Applying Graph Reduction Techniques for Identifying Structural Conflicts in Process Models. In M. Jarke and A. Oberweis, editors, *Proceedings of the 11th International Conference on Advanced Information Systems Engineering (CAiSE '99)*, volume 1626 of *Lecture Notes in Computer Science*, pages 195–209. Springer-Verlag, Berlin, 1999.
68. W. Sadiq and M.E. Orlowska. Analyzing Process Models using Graph Reduction Techniques. *Information Systems*, 25(2):117–134, 2000.
69. K. Salimifard and M. Wright. Petri Net-Based Modelling of Workflow Systems: An Overview. *European Journal of Operational Research*, 134(3):664–676, 2001.
70. R. van der Toorn. *Component-Based Software Design with Petri nets: An Approach Based on Inheritance of Behavior*. PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 2004.
71. J. Vanhatalo, H. Völzer, and F. Leymann. Faster and More Focused Control-Flow Analysis for Business Process Models Through SESE Decomposition. In B. Krämer, K. Lin, and P. Narasimhan, editors, *Proceedings of Service-Oriented*

Computing (ICSOC 2007), volume 4749 of *Lecture Notes in Computer Science*, pages 43–55. Springer-Verlag, Berlin, 2007.

72. H.M.W. Verbeek and W.M.P. van der Aalst. Analyzing BPEL Processes using Petri Nets. In D. Marinescu, editor, *Proceedings of the Second International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management*, pages 59–78. Florida International University, Miami, Florida, USA, 2005.
73. H.M.W. Verbeek, W.M.P. van der Aalst, and A.H.M. ter Hofstede. Verifying Workflows with Cancellation Regions and OR-joins: An Approach Based on Relaxed Soundness and Invariants. *The Computer Journal*, 50(3):294–314, 2007.
74. H.M.W. Verbeek, T. Basten, and W.M.P. van der Aalst. Diagnosing Workflow Processes using Woflan. *The Computer Journal*, 44(4):246–279, 2001.
75. H.M.W. Verbeek, M.T. Wynn, W.M.P. van der Aalst, and A.H.M. ter Hofstede. Reduction Rules for Reset/Inhibitor Nets. BPM Center Report BPM-07-13, BPMcenter.org, 2007.
76. M. Weske. *Business Process Management: Concepts, Languages, Architectures* . Springer-Verlag, Berlin, 2007.
77. S.A. White et al. Business Process Modeling Notation Specification (Version 1.0, OMG Final Adopted Specification), 2006.
78. A. Wombacher. Decentralized Consistency Checking in Cross-organizational Workflows. In *Proceedings of International Conference on e-Technology, e-Commerce and e-Service (CEC/EEE 2006)*, pages 39–46. IEEE Computer Society, 2006.
79. M.T. Wynn, W.M.P. van der Aalst, A.H.M. ter Hofstede, and D. Edmond. Verifying Workflows with Cancellation Regions and OR-joins: An Approach Based on Reset Nets and Reachability Analysis. In S. Dustdar, J.L. Faideiro, and A. Sheth, editors, *International Conference on Business Process Management (BPM 2006)*, volume 4102 of *Lecture Notes in Computer Science*, pages 389–394. Springer-Verlag, Berlin, 2006.
80. M.T. Wynn, D. Edmond, W.M.P. van der Aalst, and A.H.M. ter Hofstede. Achieving a General, Formal and Decidable Approach to the OR-join in Workflow using Reset nets. In G. Ciardo and P. Darondeau, editors, *Applications and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 423–443. Springer-Verlag, Berlin, 2005.
81. M.T. Wynn, H.M.W. Verbeek, W.M.P. van der Aalst, A.H.M. ter Hofstede, and D. Edmond. Reduction Rules for Reset Workflow Nets. BPM Center Report BPM-06-25, BPMcenter.org, 2006.