Anti-alignments in Conformance Checking – The Dark Side of Process Models

Thomas Chatain[1(✉)] and Josep Carmona[2]

[1] LSV, ENS Cachan, CNRS, INRIA, Université Paris-Saclay, Cachan, France
chatain@lsv.ens-cachan.fr
[2] Universitat Politècnica de Catalunya, Barcelona, Spain
jcarmona@cs.upc.edu

Abstract. Conformance checking techniques asses the suitability of a process model in representing an underlying process, observed through a collection of real executions. These techniques suffer from the well-known state space explosion problem, hence handling process models exhibiting large or even infinite state spaces remains a challenge. One important metric in conformance checking is to asses the precision of the model with respect to the observed executions, i.e., characterize the ability of the model to produce behavior unrelated to the one observed. By avoiding the computation of the full state space of a model, current techniques only provide estimations of the precision metric, which in some situations tend to be very optimistic, thus hiding real problems a process model may have. In this paper we present the notion of *anti-alignment* as a concept to help unveiling traces in the model that may deviate significantly from the observed behavior. Using anti-alignments, current estimations can be improved, e.g., in precision checking. We show how to express the problem of finding anti-alignments as the satisfiability of a Boolean formula, and provide a tool which can deal with large models efficiently.

1 Introduction

The use of process models has increased in the last decade due to the advent of the process mining field. Process mining techniques aim at discovering, analyzing and enhancing formal representations of the real processes executed in any digital environment [1]. These processes can only be observed by the footprints of their executions, stored in form of *event logs*. An event log is a collection of traces and is the input of process mining techniques. The derivation of an accurate formalization of an underlying process opens the door to the continuous improvement and analysis of the processes within an information system.

Among the important challenges in process mining, *conformance checking* is a crucial one: to assess the quality of a model (automatically discovered or manually designed) in describing the observed behavior, i.e., the event log. Conformance checking techniques aim at characterizing four quality dimensions: fitness, precision, generalization and simplicity [2]. For the first three dimensions,

© Springer International Publishing Switzerland 2016
F. Kordon and D. Moldt (Eds.): PETRI NETS 2016, LNCS 9698, pp. 240–258, 2016.
DOI: 10.1007/978-3-319-39086-4_15

the *alignment* between the process model and the event log is of paramount importance, since it allows relating modeled and observed behavior [3].

Given a process model and a trace in the event log, an alignment provides the run in the model which mostly resembles the observed trace. When alignments are computed, the quality dimensions can be defined on top [3,4]. In a way, alignments are optimistic: although observed behavior may deviate significantly from modeled behavior, it is always assumed that the least deviations are the best explanation (from the model's perspective) for the observed behavior.

In this paper we present a somewhat symmetric notion to alignments, denoted as *anti-alignments*. Given a process model and a log, an anti-alignment is a run of the model that mostly deviates from any of the traces observed in the log. The motivation for anti-alignments is precisely to compensate the optimistic view provided by alignments, so that the model is queried to return highly deviating behavior that has not been seen in the log. In contexts where the process model should adhere to a certain behavior and not leave much exotic possibilities (e.g., banking, healthcare), the absence of highly deviating anti-alignments may be a desired property to have in the process model.

We cast the problem of computing anti-alignments as the satisfiability of a Boolean formula, and provide high-level techniques which can for instance compute the most deviating anti-alignment for a certain run length, or the shortest anti-alignment for a given number of deviations.

Using anti-alignments one cannot only catch deviating behavior, but also use it to improve some of the current quality metrics considered in conformance checking. For instance, a highly-deviating anti-alignment may be a sign of a loss in precision, which can be missed by current metrics as they bound considerably the exploration of model state space for the sake of efficiency [5].

Anti-alignments are related to the *completeness of the log*; a log is complete if it contains all the behavior of the underlying process [1]. For incomplete logs, the alternatives for computing anti-alignments grows, making it difficult to tell the difference between behavior not observed but meant to be part of the process, and behavior not observed which is not meant to be part of the process. Since there exists already some metrics to evaluate the completeness of an event log (e.g., [6]), we assume event logs have a high level of completeness before they are used for computing anti-alignments.

To summarize, the contributions of the paper are now enumerated.

- We propose the notion of anti-alignment as an effective way to explore process deviations with respect to observed behavior.
- We present an encoding of the problem of computing anti-alignments into SAT, and have implemented it in the tool DarkSider.
- We show how anti-alignments can be used to provide an estimation of precision that uses a different perspective from the current ones.

The remainder of the paper is organized as follows: in the next section, a simple example is used to emphasize the importance of computing anti-alignments.

Then in Sect. 3 the basic theory needed for the understanding of the paper is introduced. Section 4 provides the formal definition of anti-alignments, whilst Sect. 5 formalizes the encoding into SAT of the problem of computing anti-alignments and Sect. 6 presents some adaptions of the notion of anti-alignments. In Sect. 7, we define a new metric, based on anti-alignments, for estimating precision of process models. Experiments are reported in Sect. 8, and related work in Sect. 9. Section 10 concludes the paper and gives some hints for future research directions.

2 A Motivating Example

Let us use the example shown in Fig. 1 for illustrating the notion of anti-alignment. The example was originally presented in [7]. The modeled process describes a realistic transaction process within a banking context. The process contains all sort of monetary checks, authority notifications, and logging mechanisms. The process is structured as follows (Fig. 1 (top) shows a high-level overview of the complete process): it is initiated when a new transaction is requested, opening a new instance in the system and registering all the components involved. The second step is to run a check on the person (or entity) origin of the monetary transaction. Then, the actual payment is processed differently, depending of the payment modality chosen by the sender (cash, cheque and payment). Later, the receiver is checked and the money is transferred. Finally, the process ends registering the information, notifying it to the required actors and authorities, and emitting the corresponding receipt. The detailed model, formalized as a Petri net, is described in the bottom part of the figure.

Assume that a log which contains different transactions covering all the possibilities with respect of the model in Fig. 1 is given. For this pair of model and log, no highly deviating anti-alignment will be obtained since the model is a precise representation of the observed behavior. Now assume that we modify a bit the model, adding a loop around the alternative stages for the payment. Intuitively, this (malicious) modification in the process model may allow to pay several times although only one transfer will be done. The modified high-level overview is shown in Fig. 2. Current metrics for precision (e.g., [5]) will not consider this modification as a severe one: the precision of the model with respect to the log will be very similar before or after the modification.

Clearly, this modification in the process models comes with a new highly deviating anti-alignment denoting a run of the model that contains more than one iteration of the payment. This may be considered as a certification of the existence of a problematic behavior allowed by the model.

3 Preliminaries

Definition 1 ((Labeled) Petri Net). *A (labeled) Petri Net* [8] *is a tuple* $N = \langle P, T, \mathcal{F}, m_0, \Sigma, \lambda \rangle$*, where* P *is the set of places,* T *is the set of transitions*

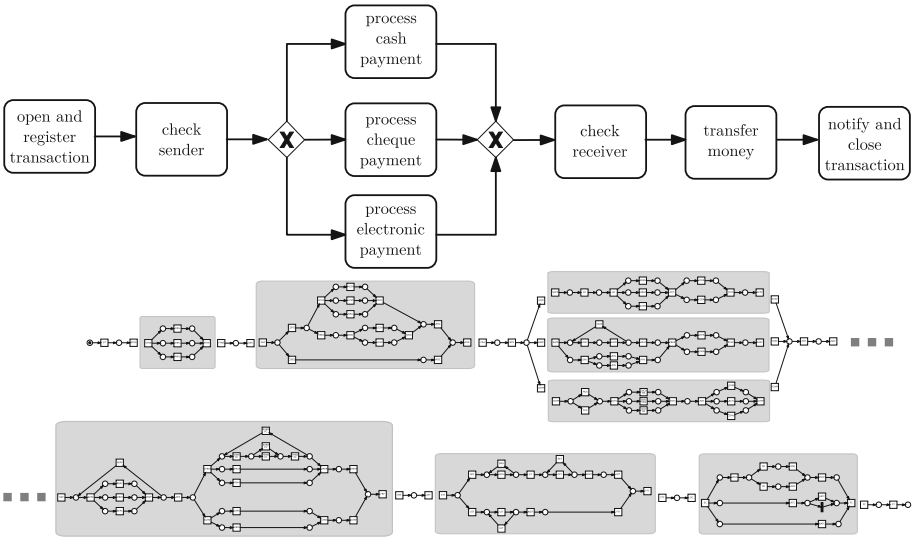


Fig. 1. Running example (adapted from [7]). Overall structure (top), process model (bottom).

(with $P \cap T = \emptyset$), $\mathcal{F} : (P \times T) \cup (T \times P) \to \{0, 1\}$ is the flow relation, m_0 is the initial marking, Σ is an alphabet of actions and $\lambda : T \to \Sigma$ labels every transition by an action.

A marking is an assignment of a non-negative integer to each place. If k is assigned to place p by marking m (denoted $m(p) = k$), we say that p is marked with k tokens. Given a node $x \in P \cup T$, we define its pre-set ${}^\bullet x := \{y \in P \cup T \mid (x, y) \in F\}$ and its post-set $x^\bullet := \{y \in P \cup T \mid (y, x) \in F\}$.

A transition t is *enabled* in a marking m when all places in ${}^\bullet t$ are marked. When a transition t is enabled, it can *fire* by removing a token from each place in ${}^\bullet t$ and putting a token to each place in $t^\bullet$. A marking m' is *reachable* from m if there is a sequence of firings $t_1 t_2 \dots t_n$ that transforms m into m', denoted by $m[t_1 t_2 \dots t_n\rangle m'$. A sequence of actions $a_1 a_2 \dots a_n$ is a *feasible sequence* (or *run*, or *model trace*) if there exists a sequence of transitions $t_1 t_2 \dots t_n$ firable from m_0 and such that for $i = 1 \dots n$, $a_i = \lambda(t_i)$. Let $\mathcal{L}(N)$ be the set of feasible sequences of Petri net N. A *deadlock* is a reachable marking for which no transition is enabled. The set of reachable markings from m_0 is denoted by $[m_0\rangle$, and form a graph called *reachability graph*. A Petri net is *k-bounded* if no marking in $[m_0\rangle$ assigns more than k tokens to any place. A Petri net is *safe* if it is 1-bounded. In this paper we assume safe Petri nets.

An event log is a collection of traces, where a trace may appear more than once. Formally:

Definition 2 (Event Log). *An* event log *L (over an alphabet of actions Σ) is a multiset of traces $\sigma \in \Sigma^*$.*

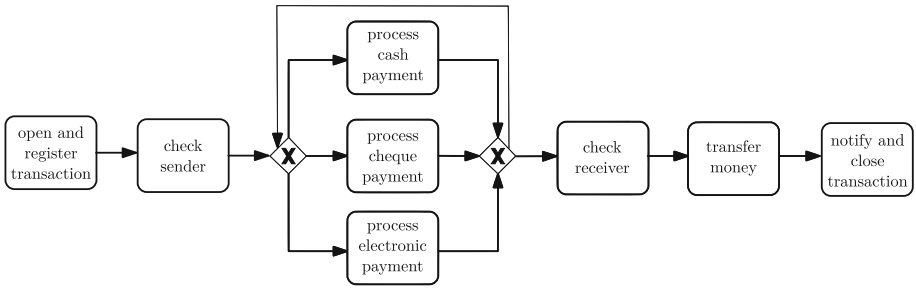


Fig. 2. Model containing a highly deviating anti-alignment for the log considered.

Quality Dimensions. Process mining techniques aim at extracting from a log L a process model N (e.g., a Petri net) with the goal to elicit the process underlying a system $\mathcal{S}$. By relating the behaviors of L, $\mathcal{L}(N)$ and $\mathcal{S}$, particular concepts can be defined [9]. A log is *incomplete* if $\mathcal{S}\backslash L \neq \emptyset$. A model N *fits* log L if $L \subseteq \mathcal{L}(N)$. A model is *precise* in describing a log L if $\mathcal{L}(N)\backslash L$ is small. A model N represents a *generalization* of log L with respect to system $\mathcal{S}$ if some behavior in $\mathcal{S}\backslash L$ exists in $\mathcal{L}(N)$. Finally, a model N is *simple* when it has the minimal complexity in representing $\mathcal{L}(N)$, i.e., the well-known *Occam's razor principle*.

4 Anti-alignments

The idea of anti-alignments is to seek in the language of a model N what are the runs which differ a lot with all the observed traces. For this we first need a definition of distance between two traces (typically a model trace, i.e. a run of the model, and an observed log trace). Relevant definitions about alignments can be found in [3]. Let us start here with a simple definition. We will discuss other definitions in Sect. 6.

Definition 3 (Hamming Distance *dist*). *For two traces $\gamma = \gamma_1 \dots \gamma_n$ and $\sigma = \sigma_1 \dots \sigma_n$, of same length n, define $dist(\gamma, \sigma) := \big|\{i \in \{1 \dots n\} \mid \gamma_i \neq \sigma_i\}\big|$.*

Definition 4. *In order to deal with traces of different length, we define for every trace $\sigma = \sigma_1 \dots \sigma_p$ and $n \in \mathbb{N}$, the trace $\sigma_{|1\dots n}$ as:*

- *$\sigma_{|1\dots n} := \sigma_1 \dots \sigma_n$, i.e. the trace σ truncated to length n, if $|\sigma| \geq n$,*
- *$\sigma_{|1\dots n} := \sigma_1 \dots \sigma_p \cdot w^{n-p}$, i.e. the trace σ padded to length n with the special symbol $w \notin \Sigma$ (w for 'wait'), if $|\sigma| \leq n$.*

Notice that the two definitions coincide when $p = n$ and give $\sigma_{|1\dots n} := \sigma$.

In the sequel, we write $dist(\gamma, \sigma)$ for $dist(\gamma, \sigma_{|1\dots|\gamma|})$.

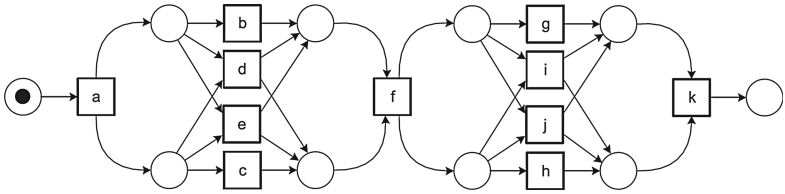


Fig. 3. The process model (taken from [10]) has the anti-alignment $\langle a, b, c, f, i, k\rangle$ for the log $L = \{\langle a, b, c, f, g, h, k\rangle, \langle a, c, b, f, g, h, k\rangle, \langle a, c, b, f, h, g, k\rangle, \langle a, b, c, f, h, g, k\rangle, \langle a, e, f, i, k\rangle, \langle a, d, f, g, h, k\rangle, \langle a, e, f, h, g, k\rangle\}$.

Definition 5 (Anti-alignment). *An (n, m)-anti-alignment of a model N w.r.t. a log L is a run $\gamma \in \mathcal{L}(N)$ such that*

- $|\gamma| = n$ *and*
- *for every $\sigma \in L$, $dist(\gamma, \sigma) \geq m$.*

Notice that, in this definition, only σ is truncated or padded. In particular this means that γ is compared to the prefixes of the observed traces. The idea is that a run γ which is close to a prefix of an observed trace is good, while a run γ which is much longer than an observed trace σ cannot be considered close to σ even if its prefix $\gamma_{|1\ldots|\sigma|}$ is close to σ.

Example 1. For instance, for the Petri net shown in Fig. 3, and the log $L = \{\langle a, b, c, f, g, h, k\rangle, \langle a, c, b, f, g, h, k\rangle, \langle a, c, b, f, h, g, k\rangle, \langle a, b, c, f, h, g, k\rangle, \langle a, e, f, i, k\rangle, \langle a, d, f, g, h, k\rangle, \langle a, e, f, h, g, k\rangle\}$, the run $\langle a, b, c, f, i, k\rangle$ denotes an $(6, 2)$-anti-alignment. Notice that for $m \geq 3$ there are no anti-alignments for this example.

Lemma 1. *If the model has no deadlock, then for every $n \in \mathbb{N}$, for every $m \in \mathbb{N}$, if there exists a (n, m)-anti-alignment γ, then there exists a $(n + 1, m)$-anti-alignment. Moreover, for $n \geq \max_{\sigma \in L} |\sigma|$, there exists a $(n + 1, m + 1)$-anti-alignment.*

Proof. It suffices to fire one transition t enabled in the marking reached after γ; $\gamma \cdot t$ is a $(n+1, m)$-anti-alignment since for every $\sigma \in L$, $dist(\gamma \cdot t, \sigma) \geq dist(\gamma, \sigma)$. When $n \geq \max_{\sigma \in L} |\sigma|$, we have more: $dist(\gamma \cdot t, \sigma) \geq 1 + dist(\gamma, \sigma)$ (because the t is compared to the padding symbol w), which makes $\gamma \cdot t$ a $(n+1, m+1)$-anti-alignment. □

Corollary 1. *If the model has no deadlock, (and assuming that the log L is a finite multiset of finite traces), then for every $m \in \mathbb{N}$, there is a least n for which a (n, m)-anti-alignment exists. This n is less than or equal to $m + \max_{\sigma \in L} |\sigma|$.*

Lemma 2. *The problem of finding a (n, m)-anti-alignment is NP-complete. (Since n and m are typically smaller than the length of the traces in the log, we assume that they are represented in unary.)*

Proof. The problem is clearly in NP: checking that a run γ is a (n, m)-anti-alignment for a net N and a log L takes polynomial time.

For NP-hardness, we propose a reduction from the problem of reachability of a marking M in a 1-safe acyclic[1] Petri net N, known to be NP-complete [11,12]. The reduction is as follows: equip the 1-safe acyclic Petri net N with complementary places[2]: a place $\overline{p}$ for each $p \in P$, with $\overline{p}$ initially marked iff p is not, $\overline{p} \in {}^\bullet t$ iff $p \in t^\bullet \setminus {}^\bullet t$, and $\overline{p} \in t^\bullet$ iff $p \in {}^\bullet t \setminus t^\bullet$. Now M is reachable in the original net iff $M \cup \{\overline{p} \mid p \in P \setminus M\}$ is reachable in the complemented net (and with the same firing sequence).

Notice that, since N is acyclic, each transition can fire only once; hence, the length of the firing sequences of N is bounded by the number of transitions $|T|$.

Add now a new transition t_f with ${}^\bullet t_f = t_f{}^\bullet = M \cup \{\overline{p} \mid p \in P \setminus M\}$. Transition t_f is firable if and only if M is reachable in the original net, and in this case, t_f may fire forever. As a consequence the new net (call it N_f) has a firing sequence of length $|T| + 1$ iff M is reachable in N.

It remains to observe that a firing sequence of length $|T| + 1$ is nothing but a $(|T| + 1, 0)$-anti-alignment for N_f and the empty log. Then M is reachable in N iff such anti-alignment exists. □

5 Computation of Anti-alignments

In order to compute a (n, m)-anti-alignment of a net N w.r.t. a log L, our tool DarkSider constructs a SAT formula $\Phi^n_m(N, L)$ and calls a SAT solver (currently minisat [13]) to solve it. Every solution to the formula is interpreted as a run of N of length n which has at least m misalignments with every log in L.

The formula $\Phi^n_m(N, L)$ characterizes a (n, m)-anti-alignment γ:

- $\gamma = \lambda(t_1) \dots \lambda(t_n) \in \mathcal{L}(N)$, and
- for every $\sigma \in L$, $dist(\gamma, \sigma) \geq m$.

5.1 Coding $\Phi^n_m(N, L)$ Using Boolean Variables

The formula $\Phi^n_m(N, L)$ is coded using the following Boolean variables:

- $\tau_{i,t}$ for $i = 1 \dots n$, $t \in T$ (remind that w is the special symbol used to pad the logs, see Definition 4) means that transition $t_i = t$.
- $m_{i,p}$ for $i = 0 \dots n$, $p \in P$ means that place p is marked in marking M_i (remind that we consider only safe nets, therefore the $m_{i,p}$ are Boolean variables).
- $\delta_{i,\sigma,k}$ for $i = 1 \dots n$, $\sigma \in L$, $k = 1, \dots, m$ means that the k^{th} mismatch with the observed trace σ is at position i.

[1] A Petri net is acyclic if the transitive closure $\mathcal{F}^+$ of its flow relation is irreflexive.
[2] In general the net does not remain acyclic with the complementary places.

The total number of variables is $n \times (|T| + |P| + |L| \times m)$.

Let us decompose the formula $\Phi^n_m(N, L)$.

– The fact that $\gamma = \lambda(t_1) \dots \lambda(t_n) \in \mathcal{L}(N)$ is coded by the conjunction of the following formulas:
  - Initial marking:

$$\left(\bigwedge_{p \in M_0} m_{0,p} \right) \wedge \left(\bigwedge_{p \in P \setminus M_0} \neg m_{0,p} \right)$$

  - One and only one t_i for each i:

$$\bigwedge_{i=1}^{n} \bigvee_{t \in T} (\tau_{i,t} \wedge \bigwedge_{t' \in T} \neg \tau_{i,t'})$$

  - The transitions are enabled when they fire:

$$\bigwedge_{i=1}^{n} \bigwedge_{t \in T} (\tau_{i,t} \implies \bigwedge_{p \in {}^\bullet t} m_{i-1,p})$$

  - Token game (for safe Petri nets):

$$\bigwedge_{i=1}^{n} \bigwedge_{t \in T} \bigwedge_{p \in t^\bullet} (\tau_{i,t} \implies m_{i,p})$$

$$\bigwedge_{i=1}^{n} \bigwedge_{t \in T} \bigwedge_{p \in {}^\bullet t \setminus t^\bullet} (\tau_{i,t} \implies \neg m_{i,p})$$

$$\bigwedge_{i=1}^{n} \bigwedge_{t \in T} \bigwedge_{p \in P, p \notin {}^\bullet t, p \notin t^\bullet} (\tau_{i,t} \implies (m_{i,p} \iff m_{i-1,p}))$$

– Now, the constraint that γ deviates from the observed traces (for every $\sigma \in L$, $dist(\gamma, \sigma) \geq m$) is coded as:

$$\bigwedge_{\sigma \in L} \bigwedge_{k=1}^{m} \bigvee_{i=1}^{n} \delta_{i,\sigma,k}$$

with the $\delta_{i,\sigma,k}$ correctly affected w.r.t. $\lambda(t_i)$ and σ_i:

$$\bigwedge_{\sigma \in L} \bigwedge_{k=1}^{m} \bigwedge_{i=1}^{n} \left(\delta_{i,\sigma,k} \iff \bigvee_{t \in T,\ \lambda(t) = \sigma_i} \tau_{i,t} \right)$$

and that for $k \neq k'$, the k^{th} and k'^{th} mismatch correspond to different i's (i.e. a given mismatch cannot serve twice):

$$\bigwedge_{\sigma \in L} \bigwedge_{i=1}^{n} \bigwedge_{k=1}^{m-1} \bigwedge_{k'=k+1}^{m} \neg(\delta_{i,\sigma,k} \wedge \delta_{i,\sigma,k'})$$

5.2 Size of the Formula

In the end, the first part of the formula ($\gamma = \lambda(t_1)\dots\lambda(t_n) \in \mathcal{L}(N)$) is coded by a Boolean formula of size $O(n \times |T| \times |N|)$, with $|N| := |T| + |P|$.

The second part of the formula (for every $\sigma \in L$, $dist(\gamma, \sigma) \geq m$) is coded by a Boolean formula of size $O(n \times m^2 \times |L| \times |T|)$.

The total size for the coding of the formula $\Phi^n_m(N, L)$ is

$$O\left(n \times |T| \times \left(|N| + m^2 \times |L|\right)\right) .$$

5.3 Solving the Formula in Practice

In practice, our tool DarkSider builds the coding of the formula $\Phi^n_m(N, L)$ using the Boolean variables $\tau_{i,t}$, $m_{i,p}$ and $\delta_{i,\sigma,k}$.

Then we need to transform the formula in conjunctive normal form (CNF) in order to pass it to the SAT solver minisat. We use Tseytin's transformation [14] to get a formula in conjunctive normal form (CNF) whose size is linear in the size of the original formula. The idea of this transformation is to replace recursively the disjunctions $\phi_1 \vee \dots \vee \phi_n$ (where the ϕ_i are not atoms) by the following equivalent formula:

$$\exists x_1, \dots, x_n \quad \begin{cases} x_1 \vee \dots \vee x_n \\ \wedge\, x_1 \implies \phi_1 \\ \wedge \dots \\ \wedge\, x_n \implies \phi_n \end{cases}$$

where $x_1, \dots, x_n$ are fresh variables.

In the end, the result of the call to minisat tells us if there exists a run $\gamma = \lambda(t_1)\dots\lambda(t_n) \in \mathcal{L}(N)$ which has at least m misalignments with every observed trace $\sigma \in L$. If a solution is found, we extract the run γ using the values assigned by minisat to the Boolean variables $\tau_{i,t}$.

5.4 Finding the Largest m for n

It follows directly from Definition 5 that, for a model N and a log L, every $(n, m+1)$-anti-alignment is also a (n, m)-anti-alignment.

Notice also that, by Definition 5, there cannot exist any $(n, n+1)$-anti-alignment and that, assuming that the model N has a run γ of length n, this run is a $(n, 0)$-anti-alignment (otherwise there is no (n, m)-anti-alignment for any m).

(Under the latter assumption), we are interested in finding, for a fixed n, the largest m for which there exists a (n, m)-anti-alignment, i.e. the run of length n of the model which deviates most from all the observed traces. Our tool DarkSider computes it by dichotomy of the search interval for m: $[0, n]$.

5.5 Finding the Least n for m

If the model N has no deadlock, then by Corollary 1, for every $m \in \mathbb{N}$, there is a least n for which a (n, m)-anti-alignment exists.

Then it is relevant to find, for a fixed m, the least n for which there exists a (n, m)-anti-alignment, i.e. (the length of) the shortest run of N which has at least m mismatches with any observed trace.

Corollary 1 tells us that the least n belongs to the interval $[m, m + \max_{\sigma \in L} |\sigma|]$. Then it can be found simply by dichotomy over this interval. However, in practice, when $\max_{\sigma \in L} |\sigma|$ is much larger than m, the dichotomy would require to check the satisfiability of $\Phi^n_m(N, L)$ for large values of n, which is costly.

Therefore our tool DarkSider proceeds as follows: it checks the satisfiability of the formulas $\Phi^m_m(N, L)$, then $\Phi^{2m}_m(N, L)$, then $\Phi^{4m}_m(N, L) \ldots$ until it finds a p such that $\Phi^{2^p m}_m(N, L)$ is satisfiable. Then it starts the dichotomy over the interval $[m, 2^p m]$.

6 Relaxations of Anti-alignments

6.1 Limiting the Use of Loops

A delicate issue with anti-alignments is to deal with loops in the model N: inserting loops in a model is a relevant way of coding the fact that similar traces were observed with a various number of iterations of a pattern. Typically, if the log contains traces ac, abc, $abbc$, ..., $abbbbbbbc$, it is fair to propose a model whose language is ab^*c.

However a model with loops necessarily generate (n, m)-anti-alignments even for large m: it suffices to take the loops sufficiently many more times than what was observed in the log. Intuitively, these anti-alignments are cheated and one does not want to blame the model for generating them, i.e., the model correctly generalizes the behavior observed in the event log. Instead, it is interesting to focus the priority on the anti-alignments which do not use the loops too often.

Our technique can easily be adapted so that it limits the use of loops when finding anti-alignments. The simplest idea is to add a new input place (call it $bound_t$) to every transition t; the number of tokens present in $bound_t$ in the initial marking determines how many times t is allowed to fire. The drawback of this trick is that the model does not remain 1-safe, and our tool currently deals only with 1-safe nets.

An alternative is to duplicate the transition t with t', $t'' \ldots$ (all labeled $\lambda(t)$) and to allow only one firing per copy (using input places $bound_t$, $bound_{t'} \ldots$ like before, but now we need only one token per place).

Finally, another way to limit the use of loops is to introduce appropriate constraints directly in the formula $\Phi^n_m(N, L)$.

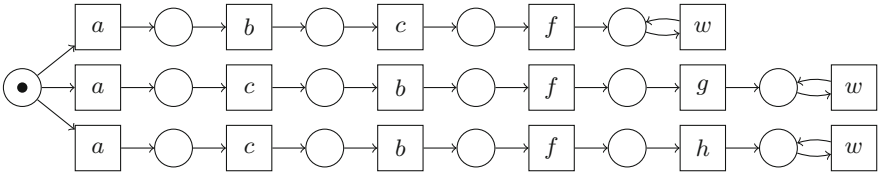


Fig. 4. The net N_L for $L = \{\langle a, b, c, f\rangle, \langle a, c, b, f, g\rangle, \langle a, c, b, f, h\rangle\}$.

6.2 Improving the Notion of Distance

A limitation of our technique as presented above, concerning the application to process mining, is that it relies on a notion of distance between γ and σ which is too rigid: indeed, every symbol of γ_i is compared only to the exact corresponding symbol σ_i. This puts for instance the word *abababababab* at distance 10 from *bababababa*. In process mining techniques often other distances are usually preferred (see for instance [3]), typically Levenshtein's distance (or edit distance), which counts how many deletions and insertions of symbols are needed to obtain σ starting from γ.

We propose here an intermediate definition where every γ_i is compared to all the σ_j for j sufficiently close to i.

Definition 6 ($dist_d$). *Let $d \in \mathbb{N}$. For two traces $\gamma = \gamma_1 \dots \gamma_n$ and $\sigma = \sigma_1 \dots \sigma_n$, of same length n, we define*

$$dist_d(\gamma, \sigma) := \left|\left\{i \in \{1 \dots n\} \mid \forall\ i - d \leq j \leq i + d \quad \gamma_i \neq \sigma_j\right\}\right|$$

Notice that $dist_0$ corresponds to the Hamming distance.

This definition is sufficiently permissive for many applications, and we can easily adapt our technique to it, simply by adapting the constraints relating the $\delta_{i,\sigma,k}$ with the $\lambda(t_i)$ in the definition of $\Phi^n_m(N, L)$.

6.3 Anti-alignments Between Two Nets

Our notion of anti-alignments can be generalized as follows:

Definition 7. *Given $n, m \in \mathbb{N}$ and two labeled Petri nets N and N' sharing the same alphabet of labels Σ, we call (n, m)-anti-alignment of N w.r.t. N', a run N of length n which is at least at distance m from every run of N'.*

Our problem of anti-alignment for a model N and a log L corresponds precisely to the problem of anti-alignment of N w.r.t. the net N_L representing all the traces in L as disjoint sequences, all starting at a common initial place end ending by a loop labeled w, like in Fig. 4.

We show below that the problem of finding anti-alignments between two nets can be reduced to solving a 2QBF formula, i.e. a Boolean formula with an alternation of quantifiers, of the form $\exists \dots \forall \dots \phi$.

Solving 2QBF formulas is intrinsically more complex than SAT formulas (Σ_2^P-complete [15] instead of NP-complete) and 2QBF solvers are usually far from being as efficient as SAT solvers.

Anyway, the notion of anti-alignments between two nets allow us to modify the net N_L in order to code a better notion of distance, for instance inserting optional wait loops at desired places in the logs. Possibly also, one can replace N_L by another net which represents a large set of runs very concisely.

2QBF solvers are usually far from being as efficient as SAT solvers. As a matter of fact, we first did a few experiments with the 2QBF encoding, but for efficiency reasons we moved to the SAT encoding. Anyway we plan to retry the 2QBF encoding in a near future, with a more efficient 2QBF solver and some optimizations, in order to benefit from the flexibility offered by the generalization of the anti-alignment problem.

2QBF Coding. Finding a (n, m)-anti-alignment of a net N w.r.t. a net N' corresponds to finding a run $\gamma \in \mathcal{L}(N)$ such that $|\gamma| = n$ and for every $\sigma \in \mathcal{L}(N')$, $dist(\gamma, \sigma) \geq m$. This is encoded by the following 2QBF formula:

$$\exists (\tau_{i,t})_{\substack{i=1\ldots n \\ t\in T}}, (m_{i,p})_{\substack{i=0\ldots n \\ p\in P}}$$
$$\forall (\tau'_{i,t'})_{\substack{i=1\ldots n \\ t'\in T'}}, (m'_{i,p'})_{\substack{i=0\ldots n \\ p'\in P'}}, (\delta_{i,k})_{\substack{i=1\ldots n \\ k=1\ldots m}}$$
$$\left\{ \begin{array}{l} \lambda(t_1)\ldots\lambda(t_n) \in \mathcal{L}(N) \\ \wedge\ \lambda'(t'_1)\ldots\lambda'(t'_n) \in \mathcal{L}(N') \\ \wedge\ \Delta \end{array} \right\} \implies \textstyle\bigwedge_{k=1}^{m} \bigvee_{i=1}^{n} \delta_{i,k}$$

where:

- the variables $\tau_{i,t}$ and $m_{i,p}$ encode the execution of N like for the coding into SAT (see Sect. 5.1); $\tau'_{i,t'}$ and $m'_{i,p'}$ represent the execution of N';
- $\delta_{i,k}$ means that the k^{th} mismatch between the two executions is at position i;
- the constraints that $\lambda(t_1)\ldots\lambda(t_n) \in \mathcal{L}(N)$ and $\lambda'(t'_1)\ldots\lambda'(t'_n) \in \mathcal{L}(N')$ are coded like in Sect. 5;
- Δ is a formula which says that the variables $\delta_{i,k}$ are correctly affected w.r.t. the values of the $\tau_{i,t}$ and $\tau'_{i,t'}$. Δ is the conjunction of:
  - there is a mismatch at the i^{th} position iff $\lambda(t_i) \neq \lambda'(t'_i)$:

$$\bigwedge_{i=1}^{n} \left((\bigvee_{k=1}^{m} \delta_{i,k}) \iff \bigvee_{\substack{t\in T, t'\in T' \\ \lambda(t)\neq\lambda'(t')}} (\tau_{i,t} \wedge \tau'_{i,t'}) \right)$$

  - a mismatch cannot serve twice:

$$\bigwedge_{k=1}^{m-1} \bigwedge_{k'=k+1}^{m} \neg(\delta_{i,k} \wedge \delta_{i,k'})$$

7 Using Anti-alignments to Estimate Precision

In this section we will provide two ways of using anti-alignments to estimate precision of process models. First, a simple metric will be presented that is based only on the information provided by anti-alignments. Second, a well-known metric for precision is introduced and it is shown how the two metrics can be combined to provide a better estimation for precision.

7.1 A New Metric for Estimating Precision

There are different ways of incorporating the information provided by anti-alignments that can help into providing a metric for precision. One possibility is to focus on the number of misalignments for a given maximal length n, i.e., find the anti-alignment with bounded length that maximizes the number of mismatches, using the search techniques introduced in the previous section. Formally, let n be the maximal length for a trace in the log, and let $max^n(N, L)$ be the maximal number of mismatches for any anti-alignment of length n for model N and log L. In practice, the length n will be set to the maximal length for a trace in the log, i.e., only searching anti-alignments that are similar in length with respect to the traces observed in the log. We can now define a simple estimation metric for precision:

$$a^n(N, L) = 1 - \frac{max^n(N, L)}{n}$$

Clearly, $max^n(N, L) \in [0 \dots n]$ which implies $a^n \in [0 \dots 1]$.

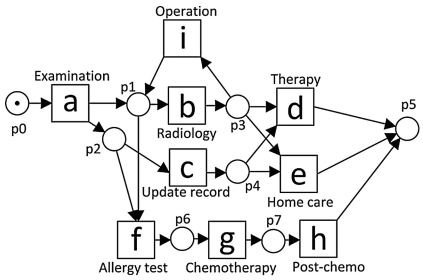
For instance, let the model be the one in Fig. 5 (top-left), and the log $L = [\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5]$ also shown in the figure. Since maximal length n for L is 6, $max^6(N, L) = 3$, corresponding to the run $\langle a, c, b, i, b, i \rangle$. Hence, $a^n = 1 - \frac{3}{6} = 0.5$.

Lemma 3 (Monotonicity of the Metric a^n). *Observing a new trace which happens to be already a run of the model, can only increase the precision measure. Formally: for every N, L and for every $\sigma \in \mathcal{L}(N)$, $a^n(N, L \cup \{\sigma\}) \geq a^n(N, L)$.*

Proof. Clearly, every (n, m)-anti-alignment for $(N, L \cup \{\sigma\})$ is also a (n, m)-anti-alignment for (N, L). Consequently $max^n(N, L \cup \{\sigma\}) \leq max^n(N, L)$ and $a^n(N, L \cup \{\sigma\}) \geq a^n(N, L)$. □

7.2 The Metric a_p

In [4,5] the metric *align precision* (a_p) was presented to estimate the precision a process model N (a Petri net) has in characterizing observed behavior, described by an event log L. Informally the computation of a_p is as follows: for each trace σ from the event log, a run γ of the model which has minimal number of deviations



trace	one-optimal	all-optimal
$\sigma_1 = \langle a \rangle$	$\langle a, f, g, h \rangle$	$\langle a, f, g, h \rangle$ $\langle a, c, b, e \rangle$ $\langle a, c, b, d \rangle$ $\langle a, b, c, d \rangle$ $\langle a, b, c, e \rangle$
$\sigma_2 = \langle a, b, c, d \rangle$	$\langle a, b, c, d \rangle$	$\langle a, b, c, d \rangle$
$\sigma_3 = \langle a, c, b, e \rangle$	$\langle a, c, b, e \rangle$	$\langle a, c, b, e \rangle$
$\sigma_4 = \langle a, f, g, h \rangle$	$\langle a, f, g, h \rangle$	$\langle a, f, g, h \rangle$
$\sigma_5 = \langle a, b, i, b, c, d \rangle$	$\langle a, b, i, b, c, d \rangle$	$\langle a, b, i, b, c, d \rangle$

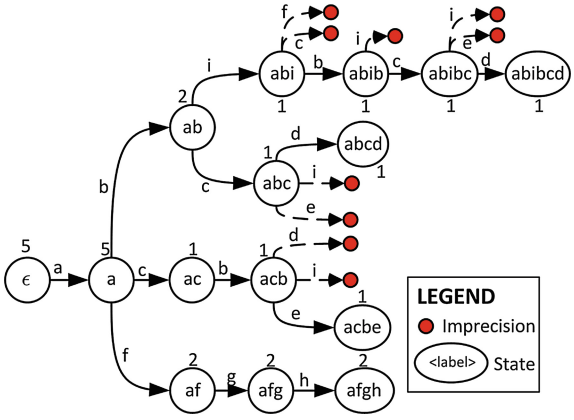


Fig. 5. Example taken from [5]. Initial process model N (top-left), optimal alignments for the event log $L = [\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5]$ (top-right), automaton $\mathcal{A}_{\Gamma(N,L)}$ (bottom).

with respect to σ is computed (denoted by $\gamma \in \Gamma(N, \sigma)$), by using the techniques from [3][3]. Let

$$\Gamma(N, L) := \bigcup_{\sigma \in L} \Gamma(N, \sigma)$$

be the set of model traces optimally aligned with traces in the log. An automaton $\mathcal{A}_{\Gamma(N,L)}$ can be constructed from this set, denoting the model's representation of the behavior observed in L. Figure 5 describes an example of this procedure. Notice that each state in the automaton has a number denoting the weight, directly related to the frequency of the corresponding prefix, e.g., in the automaton of Fig. 5, $w(ab) = 2$ and $w(acb) = 1$.

For each state s in $\mathcal{A}_{\Gamma(N,L)}$, let $a_v(s)$ be the set of available actions, i.e., possible direct successor activities according to the model, and $e_x(s)$ be the set of executed actions, i.e., activities really executed in the log. Note that, by construction $e_x(s) \subseteq a_v(s)$, i.e., the set of executed actions of a given state is

[3] Note that more than one run of the model may correspond to an optimal alignment with log trace σ, i.e., $|\Gamma(N, \sigma)| \geq 1$. For instance, in Fig. 5 five optimal alignments exist for trace $\langle a \rangle$. For the ease of explanation, we assume that $|\Gamma(N, \sigma)| = 1$.

always a subset of all available actions according to the model. By comparing these two sets in each state the metric a_p can be computed:

$$a_p(\mathcal{A}_{\Gamma(N,L)}) = \frac{\sum_{s \in Q} \omega(s) \cdot |e_x(s)|}{\sum_{s \in Q} \omega(s) \cdot |a_v(s)|}$$

where Q is the set of states in $\mathcal{A}_{\Gamma(N,L)}$. This metric evaluates to 0.780 for the automaton of Fig. 5.

Drawbacks of the Metric a_p. A main drawback of metric a_p relies in the fact that it is "short-sighted", i.e., only one step ahead of log behavior is considered in order to estimate the precision of a model. Graphically, this is illustrated in the automaton of Fig. 5 by the red states being successors of white states.

A second drawback is the lack of monotonicity, a feature that metric a^n has: observing a new trace which happens to be described by the model may unveil a model trace which has a large number of escaping arcs, thus lowering the precision value computed by a_p.

For instance, imagine that in the example of Fig. 5, the model has another long branch starting as a successor of place p_0 and allowing a large piece of behaviour. Imagine that this happens to represent a possible behaviour of the real system; simply, it has not been observed yet. This branch starting at p_0 generates a new escaping arc from the initial state of $\mathcal{A}_{\Gamma(N,L)}$, but the metric a_p does not blame a lot for this: only one more escaping point.

Now, when a trace σ corresponding to the new behaviour is observed (proving somehow that the model was right!): after this observation, the construction $\mathcal{A}_{\Gamma(N,L\cup\{\sigma\})}$ changes dramatically because it integrates the new observed trace. In consequence, if the corresponding branch in the model enables other transitions, then the model is going to be blamed for many new escaping points while, before observing σ, only one escaping point was counted.

7.3 Combining the Two Metrics

In spite of the aforementioned problems, metric a_p has proven to be a reasonable metric for precision in practice. Therefore the combination of the two metrics can lead to a better estimation of precision: whilst a_p focuses globally to count the number of escaping points from the log behavior, a^n focuses on searching globally the maximal deviation one of those escaping points can lead to.

$$a_p^n(N,L) = \alpha \cdot a_p(\mathcal{A}_{\Gamma(N,L)}) - \beta \cdot a^n(N,L)$$

with $\alpha, \beta \in \mathbb{R}_{\geq 0}$, $\alpha + \beta = 1$. Let us revisit the example introduced at the beginning of this section, which is a transformation of the model in Fig. 5 but that contains an arbitrary number of operations before the Post-chemo. If $\beta = 0.2$, then a_p^n will evaluate to 0.508, a mid value that may explicit the precision problem represented by the anti-alignment computed.

8 Experiments

We have implemented a prototype tool called DarkSider which implements the techniques described in this paper[4] Given a Petri net N and a log L, the tool is guided towards the computation of anti-alignments in different settings:

- Finding an anti-alignment of length n with at least m mismatches ($\Phi^n_m(N, L)$).
- Finding the shortest anti-alignment necessary for having at least m mismatches ($\Phi_m(N, L)$).
- Finding the anti-alignment of length n with maximal mismatches ($\Phi^n(N, L)$).

Results are provided in Table 1. We have selected two considerably large models, initially proposed in [7,16]. The table shows the size of the models (number of places and transitions), the number of traces in the log and the size of the alphabet of the log. Then the column labeled as n establishes the length imposed for the derived anti-alignment. In this columns values always start with the maximal length of a trace in the corresponding log e.g., for the first log of the prAm6 benchmark the length of any trace is less or equal to 41. Then the column m determines the minimal number of mismatches the computed anti-alignment should have. Finally, the results on computing the three formulas described above on these parameters are provided. For $\Phi^n_m(N, L)$, it is reported whereas the formula holds. For $\Phi_m(N, L)$, it is provided the length of the shortest anti-alignment found for the given number of mismatches (m). Finally, for $\Phi^n(N, L)$ we provide the number of mismatches computed for the given length (n).

For each benchmark, two different logs were used: one containing most of the behavior in the model, and the same log but where cases describing some important branch in the process model are removed. The results clearly show that using anti-alignments highly deviating behavior can be captured, e.g., for the benchmark prAm6 a very deviating anti-alignment (39 mismatches out of 41) is computed when the log does not contains that behavior in the model, whereas less deviating anti-alignments can be found for the full log (19 mismatches out of 41)[5].

9 Related Work

The seminal work in [2] was the first one in relating observed behavior (in form of a set of traces), and a process model. In order to asses how far can the model deviate from the log, the *follows* and *precedes* relations for both model and log are computed, storing for each relation whereas it *always* holds or only *sometimes*. In case of the former, it means that there is more variability. Then, log and model follows/precedes matrices are compared, and in those matrix cells where

[4] The tool is available at http://www.lsv.ens-cachan.fr/~chatain/darksider.

[5] Since in the current implementation we do not incorporate techniques for dealing with the improved distance as explained in Sect. 5, we still get a considerably deviating anti-alignment for the original log.

Table 1. Experiments for different models and logs.

benchmark	$\|P\|$	$\|T\|$	$\|L\|$	$\|A_L\|$	n	m	$\Phi^n_m(N,L)$	$\Phi_m(N,L)$	$\Phi^n(N,L)$
prAm6	347	363	761	272	41	1	✓	3	39
					41	5	✓	7	
					21	1	✓	3	19
					21	5	✓	7	
			1200	363	41	1	✓	4	19
					41	5	✓	8	
					21	1	✓	4	15
					21	5	✓	8	
BankTransfer	121	114	989	101	51	1	✓	8	32
					51	10	✓	17	
					21	1	✓	8	14
					21	10	✓	17	
			2000	113	51	1	✓	15	16
					51	10	✓	37	
					21	1	✓	15	5
					21	10	✗	37	

the model has a *sometimes* relation whilst the log has an *always* relation indicate that the model allows for more behavior, i.e., a lack of precision. This technique has important drawbacks: first, it is not general since in the presence of loops in the model the characterization of the relations is not accurate [2]. Second, the method requires a full state-space exploration of the model in order to compute the relations, a stringent limitation for models with large or even infinite state spaces.

In order to overcome the limitations of the aforementioned technique, a different approach was proposed in [4]. The idea is to find *escaping arcs*, denoting those situations where the model starts to deviate from the log behavior, i.e., events allowed by the model not observed in the corresponding trace in the log. The exploration of escaping arcs is restricted by the log behavior, and hence the complexity of the method is always bounded. By counting how many escaping arcs a pair (model, log) has, one can estimate the precision of a model. Although being a sound estimation for the precision metric, it may hide the problems we are considering in this paper, i.e., models containing escaping arcs that lead to a large behavior.

Less related is the work in [17], where the introduction of *weighted artificial negative events* from a log is proposed. Given a log L, an artificial negative event is a trace $\sigma' = \sigma \cdot a$ where $\sigma \in L$, but $\sigma' \notin L$. Algorithms are proposed to weight the confidence of an artificial negative event, and they can be used to estimate the precision and generalization of a process model [17]. Like in [4], by only considering one step ahead of log/model's behavior, this technique may not catch serious precision/generalization problems.

10 Conclusions and Future Work

In this paper the new concept of anti-alignments is introduced as a way to catch deviations a process model may have with respect to observed behavior. We show how the problem of computing anti-alignments can be casted as the satisfiability of a Boolean formula, and have implemented a tool which automates this encoding. Experimental results performed on large models show the usefulness of the approach, being able to compute deviations when they exist.

This work starts a research direction based on anti-alignments. We consider that further steps are needed to address properly some important extensions. First, it would be interesting to put anti-alignments more in the context of process mining; for that it may be required that models have also defined a clear final state, and anti-alignments should be defined accordingly in this context. Also, the distance metric may be adapted to incorporate the log frequencies, and allow it to be less strict with respect to trace deviations concerning individual positions, loops, etc. Alternatives for the computation of anti-alignments will also be investigated. Finally, the use of anti-alignments for estimating the generalization of process models will be explored.

Acknowledgments. We thank Boudewijn van Dongen for interesting discussions related to this work. This work has been partially supported by funds from the Spanish Ministry for Economy and Competitiveness (MINECO), the European Union (FEDER funds) under grant COMMAS (ref. TIN2013-46181-C2-1-R).

References

1. van der Aalst, W.M.P.: Process Mining - Discovery Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
2. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008)
3. Adriansyah, A.: Aligning observed and modeled behavior. Ph.D. thesis, Technische Universiteit Eindhoven (2014)
4. Munoz-Gama, J.: Conformance checking and diagnosis in process mining. Ph.D. thesis, Universitat Politecnica de Catalunya (2014)
5. Adriansyah, A., Munoz-Gama, J., Carmona, J., van Dongen, B.F., van der Aalst, W.M.P.: Measuring precision of modeled behavior. Inf. Syst. E-Bus. Manag. **13**(1), 37–67 (2015)
6. Yang, H., van Dongen, B.F., her Hofstede, A.H.M., Wynn, M.T., Wang, J.: Estimating completeness of event logs. Technical report BPM-12-04, BPM Center (2012)
7. van den Broucke, S.K.L.M., Munoz-Gama, J., Carmona, J., Baesens, B., Vanthienen, J.: Event-based real-time decomposed conformance analysis. In: Meersman, R., Panetto, H., Dillon, T., Missikoff, M., Liu, L., Pastor, O., Cuzzocrea, A., Sellis, T. (eds.) OTM 2014. LNCS, vol. 8841, pp. 345–363. Springer, Heidelberg (2014)
8. Murata, T.: Petri nets: properties, analysis and applications. Proc. IEEE **77**(4), 541–574 (1989)

9. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Quality dimensions in process discovery: the importance of fitness, precision, generalization and simplicity. Int. J. Coop. Inf. Syst. **23**(1), 1440001 (2014)
10. van der Aalst, W.M.P., Kalenkova, A., Rubin, V., Verbeek, E.: Process discovery using localized events. In: Devillers, R., Valmari, A. (eds.) PETRI NETS 2015. LNCS, vol. 9115, pp. 287–308. Springer, Heidelberg (2015)
11. Stewart, I.A.: Reachability in some classes of acyclic Petri nets. Fundam. Inform. **23**(1), 91–100 (1995)
12. Cheng, A., Esparza, J., Palsberg, J.: Complexity results for 1-safe nets. Theor. Comput. Sci. **147**(1&2), 117–136 (1995)
13. Eén, N., Sörensson, N.: An extensible SAT-solver. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, pp. 502–518. Springer, Heidelberg (2004)
14. Tseytin, G.S.: On the complexity of derivation in propositional calculus. In: Slisenko, A. (ed.) Studies in Constructive Mathematics and Mathematical Logic, Part II. Seminars in Mathematics. Steklov Mathematical Institute, pp. 115–125 (1970) Translated from Russian: Zapiski Nauchnykh Seminarov LOMI, vol. 8, pp. 234–259 (1968)
15. Kleine Büning, H., Bubeck, U.: Theory of quantified Boolean formulas. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability: vol. 185. Frontiers in Artificial Intelligence and Applications, pp. 735–760. IOS Press, Amsterdam (2009)
16. Munoz-Gama, J., Carmona, J., van der Aalst, W.M.P.: Single-entry single-exit decomposed conformance checking. Inf. Syst. **46**, 102–122 (2014)
17. van den Broucke, S.K.L.M., Weerdt, J.D., Vanthienen, J., Baesens, B.: Determining process model precision and generalization with weighted artificial negative events. IEEE Trans. Knowl. Data Eng. **26**(8), 1877–1889 (2014)