Alarm-Based Prescriptive Process Monitoring[*]

Irene Teinemaa[1], Niek Tax[2], Massimiliano de Leoni[2],
Marlon Dumas[1], and Fabrizio Maria Maggi[1]

[1] University of Tartu, Estonia,
{irene.teinemaa,marlon.dumas,f.m.maggi}@ut.ee
[2] Eindhoven University of Technology, The Netherlands
{n.tax,m.d.leoni}@tue.nl

Abstract. Predictive process monitoring is concerned with the analysis of events produced during the execution of a process in order to predict the future state of ongoing cases thereof. Existing techniques in this field are able to predict, at each step of a case, the likelihood that the case will end up in an undesired outcome. These techniques, however, do not take into account what process workers may do with the generated predictions in order to decrease the likelihood of undesired outcomes. This paper proposes a framework for prescriptive process monitoring, which extends predictive process monitoring approaches with the concepts of alarms, interventions, compensations, and mitigation effects. The framework incorporates a parameterized cost model to assess the cost-benefit tradeoffs of applying prescriptive process monitoring in a given setting. The paper also outlines an approach to optimize the generation of alarms given a dataset and a set of cost model parameters. The proposed approach is empirically evaluated using a range of real-life event logs.

1 Introduction

Predictive process monitoring [1,2] is a family of techniques to predict the future state of ongoing cases of a business process based on event logs recording past executions thereof. A predictive process monitoring technique may provide predictions on the remaining execution time of an ongoing case, the next activity to be executed, or the final outcome of the case wrt. a set of possible outcomes. This paper is concerned with the latter type of predictive process monitoring, which we call *outcome-oriented* [3]. For example, in a lead-to-order process, an outcome-oriented predictive process monitoring technique may predict whether a case will end up in a purchase order (desired outcome) or not (undesired outcome).

Existing outcome-oriented predictive process monitoring techniques are able to predict, after each event of a case, the likelihood that the case will end up in an undesired outcome. These techniques are restricted in scope to predicting. They do not suggest nor prescribe how and when process workers should intervene in order to decrease the likelihood of undesired outcomes.

This paper proposes a framework to extend outcome-oriented predictive process monitoring techniques in order to make them prescriptive. Concretely, the proposed framework extends a given outcome-oriented predictive process monitoring model with a mechanism for generating alarms that lead to interventions, which, in turn, mitigate (or altogether prevent) undesired outcomes. The proposed framework is armed with a parameterized cost model that captures, among others, the tradeoff between the cost of an intervention and the cost of an undesired outcome. Based on this cost model, the paper outlines an approach for return on investment analysis of a prescriptive process monitoring system under a configuration of cost parameters and a predictive model trained on a given dataset. Finally, the paper proposes and empirically evaluates an approach to tune the generation of alarms to minimize the expected cost for a given dataset and set of parameters.

The paper is structured as follows. Section 2 introduces basic concepts and notations. Next, Section 3 presents the prescriptive process monitoring framework, Section 4 outlines the approach to optimize the alarm generation mechanism, and Section 5 reports on our empirical evaluation. Finally, Section 6 discusses related work, Section 7 delineates the limitations of our framework and consequent future work, and Section 8 summarizes the contributions.

2 Background: Events, Traces, and Event Logs

For a given set A, A^* denotes the set of all sequences over A and $\sigma = \langle a_1, a_2, \ldots, a_n \rangle$ a sequence of length n; $\langle \rangle$ is the empty sequence and $\sigma_1 \cdot \sigma_2$ is the concatenation of sequences σ_1 and σ_2. $hd^k(\sigma) = \langle a_1, a_2, \ldots, a_k \rangle$ is the prefix of length k $(0 < k < n)$ of sequence σ. For example, $hd^2(\langle a, b, c, d, e \rangle) = \langle a, b \rangle$.

Let $\mathcal{E}$ be the event universe, i.e., the set of all possible event identifiers, and $\mathcal{T}$ the time domain. We assume that events are characterized by various properties, e.g., an event has a timestamp, corresponds to an activity, is performed by a particular resource, etc. We do not impose a specific set of properties, however, we assume that two of these properties are the timestamp and the activity of an event, i.e., there is a function $\pi_{\mathcal{T}} \in \mathcal{E} \to \mathcal{T}$ that assigns timestamps to events, and a function $\pi_{\mathcal{A}} \in \mathcal{E} \to \mathcal{A}$ that assigns to each event an activity from a finite set of process activities $\mathcal{A}$. An *event log* is a set of events, each linked to one trace and globally unique, i.e., the same event cannot occur twice in a log. A trace in a log represents the execution of one case.

Definition 1 (Trace, Event Log). *A* trace *is a finite non-empty sequence of events $\sigma \in \mathcal{E}^*$ such that each event appears only once and time is non-decreasing, i.e., for $1 \leq i < j \leq |\sigma| : \sigma(i) \neq \sigma(j)$ and $\pi_{\mathcal{T}}(\sigma(i)) \leq \pi_{\mathcal{T}}(\sigma(j))$. An* event log *is a set of traces $L \subset \mathcal{E}^*$ such that each event appears at most once in the entire log.*

3 Prescriptive Process Monitoring Framework

In this section, we introduce a cost model for alarm-based prescriptive process monitoring and illustrate this model using three scenarios (Section 3.1). We then

formalize the concept of alarm system (Section 3.2) and discuss conditions under which an alarm system has a positive return on investment (Section 3.3).

3.1 Concepts and Cost Model

An alarm-based prescriptive process monitoring system (*alarm system* for short) is a monitoring system that raises an alarm in relation to a running case of a business process, in order to indicate that the case is likely to lead to an undesired outcome. These alarms are handled by process workers who intervene by performing an action (e.g., calling a customer or blocking a credit card) in order to prevent or mitigate the undesired outcome. These actions may have a cost, which we call *cost of intervention*. Instead, if the case ends in a negative outcome, this leads to a cost called *cost of undesired outcome.*

As an example, consider a municipality that needs to collect city taxes. If the inhabitants do not pay their taxes on time, the municipality may run into cash flow issues. Accordingly, in case of an unpaid tax debt (undesired outcome), the municipality may decide to outsource the debt collection to an external collection agency, for which it has to pay a recovery fee. These fees constitute the cost of the undesired outcome. In light of their characteristics and past payment history, certain inhabitants may have a higher risk of missing the payment deadline. Therefore, sending a reminder letter to these high-risk inhabitants may increase the likelihood of receiving the payment on time. However, such an intervention comes with costs related to preparing the letter by an employee (proportional to the employee's hourly salary rate) and the postal costs for sending the letter.

In certain scenarios, the cost of an intervention may increase over time, acknowledging the importance of alarming as early as possible. For instance, in a railway maintenance process, if an alarm about a possible railway disruption is raised early, the problem could be solved with regular maintenance procedures. Conversely, if the alarm is raised when the need for maintenance has become urgent, the maintenance provider could be required to allocate more resources in order to solve the problem on time.

When an alarm is raised, there is a certain probability, but no certainty, that the case will reach an undesired outcome if no intervention is made. If the case does not conclude with an undesired outcome even without interventions, doing the intervention causes unnecessary costs (e.g., a company could lose customers and/or opportunities). The cost related to such unnecessary interventions is referred to as *cost of compensation.* For instance, financial institutions may block credit card payments when they suspect that a card was cloned. However, in some cases, it may happen that the suspicion was unfounded and that the payment was legitimate. If these cases become too frequent, the reputation of the financial institution could be hampered.

The purpose of alarming is to avoid an undesired outcome. However, in several scenarios, it is not possible to fully prevent the cost of the undesired outcome, while the intervention could still help to mitigate it. Based on this rationale, we introduce the concept of *mitigation effectiveness* of an intervention, reflecting the proportion of the cost of an undesired outcome that can be avoided by carrying out the intervention. Oftentimes, the mitigation effectiveness decreases with

time, i.e., the earlier the intervention takes place, the higher is the proportion of costs that can be avoided. Consider, for instance, the process of paying unemployment benefits by a social security institution. In this case, the aim of an alarm system could be to notify the institution about citizens who might be receiving unentitled benefits. Since the benefits that have already been issued are unlikely to be recollected, the cost of the undesired outcome cannot be avoided completely. Therefore, it is important to raise the alarm as early as possible, in order to effectively mitigate the cost of the undesired outcome.

An alarm system is intended as a system where cases are continuously monitored. However, since continuous monitoring is impractical, we assume that cases are monitored after each executed event and, therefore, alarms can only be raised after that an event has occurred. In the remainder, each case is identified by a trace σ that is (eventually) recorded in an event log. Definition 2 formalizes the costs defined above. Since costs may depend on the position in the case in which the alarm is raised and/or on other cases being executed, we define the costs as functions over the number of already executed events and over the entire set of cases under execution.

Definition 2 (Alarm-based Cost Model). *An* alarm-based cost model *is a tuple* $(c_{in}, c_{out}, c_{com}, \textit{eff})$ *consisting of:*

- *a function* $c_{in} \in \mathbb{N} \times \mathcal{E}^* \times 2^{\mathcal{E}^*} \to \mathbb{R}_0^+$ *modeling the* cost of <u>in</u>tervention*: given a trace* σ *belonging to an event log* L, $c_{in}(k, \sigma, L)$ *indicates the cost of an intervention in* σ *when the intervention takes place after the k-th event;*
- *a function* $c_{out} \in \mathcal{E}^* \times 2^{\mathcal{E}^*} \to \mathbb{R}_0^+$ *modeling the* cost of undesired <u>out</u>come*;*
- *a function* $c_{com} \in \mathcal{E}^* \times 2^{\mathcal{E}^*} \to \mathbb{R}_0^+$ *modeling the* cost of <u>com</u>pensation*;*
- *a function* $\textit{eff} \in \mathbb{N} \times \mathcal{E}^* \times 2^{\mathcal{E}^*} \to [0, 1]$ *modeling the* mitigation <u>eff</u>ectiveness *of an intervention: given a trace* σ *belonging to an event log* L, $\textit{eff}(k, \sigma, L)$ *indicates the mitigation effectiveness of an intervention in* σ *when the intervention takes place after the k-th event.*

To illustrate the versatility of the above cost model, we discuss three use cases for alarm systems and their corresponding cost model configurations. The first scenario, in Box 1, refers to the provision of unemployment benefits. The cost model for this scenario is based on several discussions with the stakeholders of a real social security institution [4]. The second scenario, in Box 2, refers to the detection of malicious credit card payments in a financial institution. Differently from the previous scenario, in this case, there is a risk of cost of compensation: due to the inconvenience caused by blocking their credit card, customers can switch to competitors. Box 3 refers to the process of predictive maintenance in railway services. This scenario is different from the previous ones because, in this case, the cost of an intervention increases over time.

3.2 Alarm-Based Prescriptive Process Monitoring System

An alarm-based prescriptive process monitoring system is driven by the outcome of the cases. Hereon, the outcome of the cases is represented by a function $out \in \mathcal{E}^* \to \{\textsf{true}, \textsf{false}\}$: given a case identified by a trace σ, if the case has an undesired outcome, $out(\sigma) = \textsf{true}$; otherwise, $out(\sigma) = \textsf{false}$. In reality, during

the execution of a case, its outcome is not yet known and needs to be estimated based on past executions that are recorded in an event log $L \subset \mathcal{E}^*$. The outcome estimator is a function $\widehat{out}_L \in \mathcal{E}^* \to [0, 1]$ predicting the likelihood $\widehat{out}_L(\sigma')$ that the outcome of a case that starts with prefix σ' is undesired. We can define an

Table 1: Cost of a case σ based on its outcome and whether an alarm was raised

	undesired outcome	desired outcome
alarm raised	$c_{in}(k, \sigma, L) + (1 - \mathit{eff}(k, \sigma, L))c_{out}(\sigma, L)$	$c_{in}(k, \sigma, L) + c_{com}(\sigma, L)$
alarm not raised	$c_{out}(\sigma, L)$	0

alarm system as a function that returns true or false depending on whether an alarm is raised based on the predicted outcome or not.

Definition 3 (Alarm-Based Prescriptive Process Monitoring System). *Given an event log $L \subset \mathcal{E}^*$, let $\widehat{out}_L$ be an outcome estimator built from L. An alarm-based prescriptive process monitoring system is a function $alarm_{\widehat{out}_L} \in \mathcal{E}^* \to \{\mathsf{true}, \mathsf{false}\}$. Given a running case identified by a trace σ and with current prefix σ', $alarm_{\widehat{out}_L}(\sigma)$ returns true, if an alarm is raised based on the predicted outcome $\widehat{out}_L(\sigma')$, or false, otherwise.*

For simplicity, we omit the subscript L from $\widehat{out}_L$ and omit $\widehat{out}_L$ from $alarm_{\widehat{out}_L}$ when it is clear from the context. An alarm system can raise an alarm at most once per case, since we assume that already the first alarm triggers an intervention by the stakeholders.

The purpose of an alarm system is to minimize the cost of executing a case. Table 1 summarizes how the cost of a case is determined based on a cost model (cf. Def. 2), on the case outcome, and on whether an alarm was raised or not.

Definition 4 (Cost of Case Execution). *Let $cm = (c_{in}, c_{out}, c_{com}, \mathit{eff})$ be an alarm-based cost model. Let $out \in \mathcal{E}^* \to \{\mathsf{true}, \mathsf{false}\}$ be an outcome function. Let $alarm \in \mathcal{E}^* \to \{\mathsf{true}, \mathsf{false}\}$ be an alarm-based prescriptive process monitoring system. Let $L \subset E^*$ be the entire set of* complete *(i.e., no more running) cases. Let $\sigma \in L$ be a case. Let $\mathcal{I}(\sigma, alarm)$ be the index of the event in σ when the alarm was raised or zero if no alarm was raised:*

$$\mathcal{I}(\sigma, alarm) = \begin{cases} 0 & \text{if } \forall k \in [1, |\sigma|-1]. \neg alarm(hd^k(\sigma)), \\ 1 & \text{if } alarm(hd^1(\sigma)), \\ i \in [2, |\sigma|] \text{ s.t. } alarm(hd^i(\sigma)) \wedge & \text{otherwise.} \\ \quad \forall k \in [1, i-1]. \neg alarm(hd^k(\sigma)) \end{cases}$$

The cost of execution of case σ *supported by the alarm system is:*

$$cost(\sigma, L, cm, alarm) = \begin{cases} c_{in}(\mathcal{I}(\sigma, alarm), \sigma, L) + (1 - \textit{eff}(\mathcal{I}(\sigma, alarm), \sigma, L)) \cdot c_{out}(\sigma, L) & out(\sigma) \wedge \mathcal{I}(\sigma, alarm) > 0, \\ c_{in}(\mathcal{I}(\sigma, alarm), \sigma, L) + c_{com}(\sigma, L) & \neg out(\sigma) \wedge \mathcal{I}(\sigma, alarm) > 0, \\ c_{out}(\sigma, L) & out(\sigma) \wedge \mathcal{I}(\sigma, alarm) = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Section 4 illustrates how an alarm-based prescriptive process monitoring system can be designed aiming at the minimization of the case execution costs (according to Def. 4).

3.3 Return on Investment Analysis

In this section, we provide an analysis and guidelines that suggest when it is valuable to invest in developing an alarm system, namely, when the return on investment (ROI) is positive. To this aim, we need to compare the case of a business process execution supported by an alarm system with the *as-is* situation where the business process is executed without this support. For this analysis, we consider a set of cases recorded in an event log L, where no interventions were done, and a cost model $cm = (c_{in}, c_{out}, c_{com}, \textit{eff})$.

The *as-is* situation implies that no interventions are done in any of the cases $\sigma \in L$ that lead to an undesired outcome, yielding a cost $c_{out}(\sigma)$. When applied to the entire log L, the cost is $cost_{as\text{-}is}(L) = \sum_{\sigma \in L \text{ s.t. } out(\sigma)} c_{out}(\sigma)$. Instead, when a certain system *alarm* is in effect, the costs are $cost_{alarm}(L) = \sum_{\sigma \in L} cost(\sigma, L, cm, alarm)$ (cf. Defs. 2, 3). With this setting, the return on investment of the system *alarm* is $ROI(L, cm, alarm) = cost_{as\text{-}is}(L) - cost_{alarm}(L)$, which must be positive to make deploying the system worthwhile.

The question that remains is: *how does the ROI depend on the cost model and the alarm system?* For the sake of simplicity, we assume, in this analysis, that every component of the cost model is constant. Furthermore, the initial investment costs are not considered because we assume the system to be fully operational already for a sufficiently long time, so that the the initial costs have been amortized. The above assumptions yield the following case cost:

$$cost(\sigma, L, cm, alarm) = \begin{cases} c_{in} + (1 - \textit{eff})c_{out} & out(\sigma) \wedge \mathcal{I}(\sigma, alarm) > 0, \\ c_{in} + c_{com} & \neg out(\sigma) \wedge \mathcal{I}(\sigma, alarm) > 0, \\ c_{out} & out(\sigma) \wedge \mathcal{I}(\sigma, alarm) = 0, \\ 0 & \text{otherwise} \end{cases}$$

where c_{in}, c_{out}, c_{com}, and $\textit{eff}$ are constants. In order for the ROI to be positive, it should be $cost_{as\text{-}is}(L) > cost(\sigma, L, cm, alarm)$, that is:

$$|L_{und}| \cdot c_{out} > |L_{und\&al}|(c_{in} + (1 - \textit{eff})c_{out}) + |L_{des\&al}|(c_{in} + c_{com}) + |L_{und\&nal}| \cdot c_{out}$$

where $L_{und\&al}$, $L_{des\&al}$, $L_{und\&nal}$ respectively consist of the traces in L related to the cases with an <u>und</u>esired outcome that would be <u>al</u>armed, with a <u>des</u>ired

outcome that would still be <u>al</u>armed, with an <u>undesired</u> outcome that would <u>not</u> be <u>al</u>armed; also, $L_{und} = L_{und\&al} \cup L_{und\&nal}$. After simplification:

$$|L_{und\&al}|(\mathit{eff}\, c_{out} - c_{in}) > |L_{des\&al}|(c_{in} + c_{com}). \qquad (1)$$

Because the right-hand side of Eq. 1 is non-negative, it follows as a corollary that $\mathit{eff}\, c_{out} > c_{in}$ is a necessary condition for return on investment. In other words, it must be possible to avoid a cost that is higher than the cost of doing the intervention. This provides a validation of our framework: it complies with the *reasonableness condition* in the cost-sensitive learning literature [5], which states that the cost of labeling an example incorrectly should always be greater than the cost of labeling it correctly.

Eq. 1 also illustrates that the policy of always alarming does not yield a positive ROI, unless the number of cases with undesired outcome and the cost of the undesired outcome are sufficiently high. When the number of cases with an undesired outcome is small (e.g., the unemployment benefits and the financial institution scenarios described in Boxes 1 and 2) and at the same time the cost of this undesired outcome is small, then the left-hand side of Eq. 1 is negligible, thus leading to condition $c_{in} + c_{com} < 0$, which can never hold.

So far we have assumed, for the sake of simplicity, that costs and mitigation effectiveness are constant, similarly to traditional cost-sensitive learning. However, the novelty of our formulation lays in the fact that costs are functions that depend on the time when an intervention is made. As a result, the reasonableness of the cost matrix would not be fixed, but potentially changes over time. Still, variable costs do not invalidate the ROI analysis. In fact, in order for the ROI to be positive, it is sufficient that the cost model is reasonable for a certain time period; otherwise, the alarm system would never raise alarms because of the cost model. Clearly, the longer the reasonable-cost period is, the higher the ROI.

4   Alarming Mechanisms and Empirical Thresholding

An alarm system needs two components to minimize the costs of future cases: (1) a probabilistic classifier $\widehat{out}_L \in \mathcal{E}^* \rightarrow [0, 1]$ that estimates the likelihood of an undesired outcome for a partial trace based on some historical observations L, and (2) an alarming mechanism that, for a given incomplete case, decides whether or not to raise an alarm based on the prediction made by $\widehat{out}_L$. We propose to implement the second component using a function $agent \in [0, 1] \rightarrow \{\mathsf{true}, \mathsf{false}\}$ that operates on the estimated likelihood of an undesired outcome, where value true represents the decision to raise an alarm. Together, the two components form an *alarm system*, $alarm(hd^k(\sigma)) = agent(\widehat{out}_L(hd^k(\sigma)))$, which makes the decision on whether or not to raise an alarm based on the observed k events of trace σ.

The first component, function $\widehat{out}_L$, can be implemented using any classification algorithm that is naturally probabilistic, i.e., that outputs likelihood scores on a $[0, 1]$-interval instead of a binary outcome. Examples of probabilistic classification algorithms include naive Bayes, logistic regression, and random forest. The classifier is trained on historical cases recorded in a log L_{train}.

It is easy to see that the decision on whether or not to raise an alarm should be dependent not only on $\widehat{out}_{L_{train}}(hd^k(\sigma))$, but also on the configuration of c_{in}, c_{out}, c_{com}, and eff. When c_{in} and c_{com} are very low compared to c_{out}, it might be beneficial to use a lower threshold for the estimated likelihood $\widehat{out}_{L_{train}}(hd^k(\sigma))$, while one would want to be more certain that the undesired outcome will happen when c_{in} or c_{com} is high.

We propose to implement the second component, *agent*, as an *alarming threshold*, i.e., a mechanism that alarms when the estimated likelihood of an undesired outcome is at least τ. We define function $alarm_\tau(hd^k(\sigma))$ to be the alarming function that uses the alarming mechanism $agent_\tau(\widehat{out}_{L_{train}}(hd^k(\sigma))) = \widehat{out}_{L_{train}}(hd^k(\sigma)) \geq \tau$. We aim at finding the optimal value $\bar{\tau}$ of the alarming threshold that minimizes the cost on a log L_{thres} consisting of historical observations such that $L_{thres} \cap L_{train} = \emptyset$ with respect to a given likelihood estimator $\widehat{out}_{L_{train}}$ and cost model cm. The total cost of an alarming mechanism $alarm$ on a log L is defined as $cost(L, cm, alarm) = \Sigma_{\sigma \in L} cost(\sigma, L, cm, alarm)$. Using this definition, we define $\bar{\tau} = \arg\min_{\tau \in [0,1]} cost(L_{thres}, cm, alarm_\tau)$. Optimizing a threshold τ on a separate thresholding set is called *empirical thresholding* [6] and the search for the optimal threshold $\bar{\tau}$ wrt. a specified cost model and log L_{thres} can be performed using any hyperparameter optimization technique, such as Tree-structured Parzen Estimator (TPE) optimization [7]. The resulting approach can be considered to be a form of cost-sensitive learning, since the value $\bar{\tau}$ depends on how the cost model cm is specified.

Note that as an alternative to a single global alarming threshold $\bar{\tau}$ it is possible to optimize a separate threshold $\overline{\tau_k}$ for each prefix length k. We experimentally found a single global threshold $\bar{\tau}$ optimized on L_{thres} to outperform separate prefix-length-dependent thresholds $\overline{\tau_k}$ optimized on L_{thres}, therefore we propose to use a single optimized threshold.

After creating the fully functional alarm system by training a classifier on L_{train} and optimizing the alarming threshold on L_{thres} for the given cost model cm, the obtained alarming function $alarm$ can be applied to the continuous stream of events coming from the executions of a business process, thereby reducing the processing costs of the running cases.

5 Evaluation

In this section, we describe the experimental setup for evaluating the proposed framework and the results of the evaluation. We address the following research questions:

RQ1 Can empirical thresholding find thresholds that consistently lead to a reduction in the average processing cost for different cost model configurations?

RQ2 Does the alarm system consistently yield a benefit over different values of the mitigation effectiveness?

RQ3 Does the alarm system consistently yield a benefit over different values of the cost of compensation?

5.1 Approaches and Baselines

We experiment with two different implementations of $\widehat{out}_{L_{train}}$ by using different well-known classification algorithms, namely, random forest (RF) and gradient boosted trees (GBT). Both classification algorithms have shown to be amongst the top performing classification algorithms on a variety of classification tasks [8,9]. We employ a single classifier approach where the features for a given prefix are obtained using the aggregation encoding [10], which has been shown to perform better than alternative encodings for event logs [3].

We apply the TPE optimization procedure for the alarming mechanism to find the optimal threshold $\overline{\tau}$. We use several fixed thresholds as baselines. First, we compare with the *as-is* situation in which alarms are never raised. Secondly, we compare with the baseline $\tau = 0$, allowing us to compare with the situation where alarms are always raised directly at the start of a case. Finally, we compare with $\tau = 0.5$ enabling the comparison with the cost-insensitive scenario that simply alarms when an undesired outcome is expected. The implementation of the approach and the experimental setup are openly available online.[3]

5.2 Datasets

For each event log, we use all available data attributes as input to the classifier. Additionally, we extract the *event number*, i.e., the index of the event in the given case, the *hour, weekday, month, time since case start*, and *time since last event*. Infrequent values of categorical attributes (occurring less than 10 times in the log) are replaced with value "other", to avoid exploding the dimensionality. Missing attributes are imputed with the respective most recent (preceding) value of that attribute in the same trace when available, otherwise with zero. Traces are cut before the labeling of the case becomes trivially known and are truncated at the 90th percentile of all case lengths to avoid bias from very long traces. We use the following datasets to evaluate the alarm system:

BPIC2017. This log records execution traces from a loan application process in a Dutch financial institution.[4] The event log was split into two sub-logs, denoted with *bpic2017_refused* and *bpic2017_cancelled*. In the first one, the undesired cases refer to the process executions in which the applicant has refused the final offer(s) by the financial institution and, in the second one, the undesired cases consist of those cases where the financial institution has cancelled the offer(s).

Road traffic fines. This event log originates from the Italian local police.[5] The desired outcome is that a fine is fully paid, while in the undesired cases the fine needs to be sent for credit collection.

Unemployment. This event log corresponds to the *Unemployment Benefits* scenario (Box 1 in Section 3.1). The undesired outcome is that a resident will receive more benefits than entitled, causing the need for a reclamation. Privacy constraints prevent us from making this event log publicly available.

---

[3] `https://taxxer.github.io/AlarmBasedProcessPrediction/`
[4] `https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b`
[5] `https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5`

Table 2: Dataset statistics

dataset name	# traces	class ratio	min length	med length	(trunc.) max length	# events
bpic2017_refused	31 413	0.12	10	35	60	1 153 398
bpic2017_cancelled	31 413	0.47	10	35	60	1 153 398
traffic_fines	129 615	0.46	2	4	5	445 959
unemployment	34 627	0.2	1	21	79	1 010 450

Table 2 describes the characteristics of the event logs used. The classes are well balanced in *bpic2017_cancelled* and *traffic_fines*, while the undesired outcome is more rare in case of *unemployment* and *bpic2017_refused*. In *traffic_fines*, the traces are very short, while in the other datasets the traces are longer.

5.3 Experimental Setup

We apply a temporal split, i.e., we order the cases by their start time and from the first 80% of the cases randomly select 80% (i.e., 64% of the total) for L_{train} and 20% (i.e., 16% of the total) for L_{thres}, and use the remaining 20% as the test set L_{test}. The events in cases in L_{train} and L_{thres} that overlap in time with L_{test} are discarded in order to not use any information that would not be available yet in a real setting. We use TPE with 3-fold cross validation on L_{train} to optimize the hyperparameters for RF and GBT. We optimize the alarming threshold $\overline{\tau}$ by building the final classifiers using all the traces in L_{train} and search for $\overline{\tau}$ using L_{thres}.

It is common in cost-sensitive learning to apply calibration techniques to the resulting classifier in order to obtain accurate probability estimates and, therefore, more accurate estimates of the expected cost [11]. However, we found that calibrating the classifier using Platt scaling [12] does not consistently improve the estimated likelihood of undesired outcome on the four event logs, and frequently even leads to less accurate likelihood estimates. Therefore, we decided to skip the calibration step. Moreover, since we use empirical thresholding, it is not necessary that the probabilities are well calibrated and it is sufficient that the likelihoods are reasonably ordered.

Table 3 shows the configurations of the cost model that we explore in the evaluation. To answer RQ1, we vary the ratio between $c_{out}(\sigma, L)$ and $c_{in}(k, \sigma, L)$ (keeping $c_{com}(\sigma, L)$ and $eff(k, \sigma, L)$ unchanged). To answer RQ2, we vary both $eff(k, \sigma, L)$ and the ratio between $c_{out}(\sigma, L)$ and $c_{in}(k, \sigma, L)$. To answer RQ3, we vary two ratios: 1) between $c_{out}(\sigma, L)$ and $c_{in}(k, \sigma, L)$ and 2) between $c_{in}(k, \sigma, L)$ and $c_{com}(\sigma, L)$.

We measure the average processing cost per case in L_{test}, and aim at minimizing this cost. Additionally, we measure the *benefit* of the alarm system, i.e., the reduction in the average processing cost of a case when using the alarm system compared to the average processing cost when not using it.

Table 3: Cost model configurations

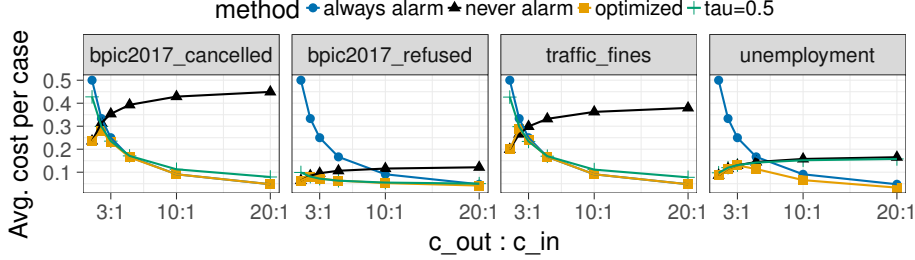|	$c_{out}(\sigma, L)$	$c_{in}(k, \sigma, L)$	$c_{com}(\sigma, L)$	$\textit{eff}(k, \sigma, L)$
RQ1	$\{1, 2, 3, 5, 10, 20\}$	1	0	$1 - k/|\sigma|$
RQ2	$\{1, 2, 3, 5, 10, 20\}$	1	0	$\{0, 0.1, 0.2, ..., 1\}$
RQ3	$\{1, 2, 3, 5, 10, 20\}$	1	$\{0, 1/20, 1/10, 1/5, 1/2, 1, 2, 5, 10, 20\}$	$1 - k/|\sigma|$

Fig. 1: Cost over different ratios of $c_{out}(\sigma, L)$ and $c_{in}(k, \sigma, L)$ (GBT)

5.4 Results

Fig. 1 shows the average cost per case when increasing the ratio of $c_{out}(\sigma, L)$ and $c_{in}(k, \sigma, L)$ from left to right. We only present the results obtained with GBT as we found it to slightly outperform RF. When the ratio between these two costs is balanced (i.e., 1:1), the minimal cost is obtained by never alarming. This is in agreement with the ROI analysis, where we found $\textit{eff}\, c_{out} > c_{in}$ to be a necessary condition for having an advantage from an alarm system. When $c_{out} \gg c_{in}$ the best strategy is to always alarm. When c_{out} is slightly higher than c_{in} the best strategy is to sometimes alarm based on $\widehat{out}$. We found that the optimized $\overline{\tau}$ always outperforms the baselines. An exception is ratio 2:1 for *traffic_fines*, where never alarming is slightly better.

In Fig. 2, the average cost per case is plotted against different (fixed) thresholds. The optimized threshold is marked with a red cross and each line represents one particular cost ratio. We observe that, while the optimized threshold generally obtains minimal costs, there sometimes exist multiple optimal thresholds for a given cost model configuration. For instance, in the case of the 5:1 ratio in *bpic2017_cancelled*, all thresholds between 0 and 0.4 are cost-wise equivalent. We conclude that the empirical thresholding approach consistently finds a threshold that yields the lowest cost in a given event log and cost model configuration (cf. RQ1).

Fig. 3a shows the benefit of having an alarm system compared to not having it for different (constant) mitigation effectiveness values. As the results are similar for logs with similar class ratios, hereinafter, we only show the results for one log from each of the groups: *bpic2017_cancelled* (balanced classes) and *unemployment* (imbalanced classes). As expected, the benefit increases both with higher $\textit{eff}(k, \sigma, L)$ and with higher $c_{out}(\sigma, L) : c_{in}(k, \sigma, L)$ ratio. For *bpic2017_cancelled*,
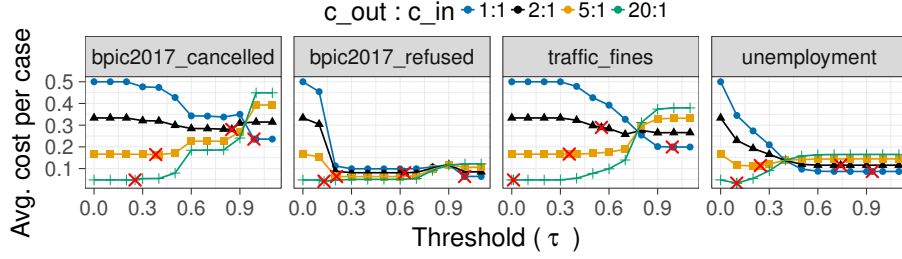
Fig. 2: Cost over different thresholds ($\bar{\tau}$ is marked with a red cross)

(a) Varying $eff(k, \sigma, L)$
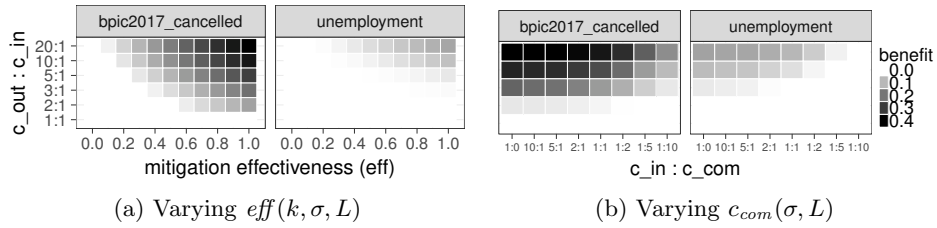
(b) Varying $c_{com}(\sigma, L)$

Fig. 3: Benefit with different cost model configurations

the alarm system yields a benefit when $c_{out}(\sigma, L) : c_{in}(k, \sigma, L)$ is high and $eff(k, \sigma, L) > 0$. Also, a benefit is always obtained when $eff(k, \sigma, L) > 0.5$ and $c_{out}(\sigma, L) > c_{in}(k, \sigma, L)$. In the case of *unemployment*, the average benefits are smaller, since there are fewer cases with undesired outcome and, therefore, the number of cases where c_{out} can be prevented by alarming is lower. In this case, a benefit is obtained when both $eff(k, \sigma, L)$ and $c_{out}(\sigma, L) : c_{in}(k, \sigma, L)$ are high. We conducted analogous experiments with linear effectiveness decay, varying the maximum possible effectiveness (at the start of the case), which confirmed that the observed patterns remain the same. We have empirically confirmed our theoretical finding (Section 3.3) that $eff\, c_{out} > c_{in}$ is a necessary condition to obtain a benefit from using an alarm system, and have shown that a benefit is in practice also obtained under this condition when an optimized alarming threshold is used (cf. RQ2).

Similarly, the benefit of the alarm system is plotted in Fig. 3b across different ratios of $c_{out}(\sigma, L) : c_{in}(k, \sigma, L)$ and $c_{in}(k, \sigma, L) : c_{com}(\sigma, L)$. We observe that when $c_{com}(\sigma, L)$ is high, the benefit decreases due to false alarms. For *bpic2017_cancelled*, a benefit is obtained almost always, except when $c_{out}(\sigma, L) : c_{in}(k, \sigma, L)$ is low (e.g., 2:1) and $c_{com}(\sigma, L)$ is high (i.e., higher than $c_{in}(k, \sigma, L)$). For *unemployment*, a benefit is obtained with fewer cost model configurations, e.g., when $c_{out}(\sigma, L) : c_{in}(k, \sigma, L) = 5 : 1$ and $c_{com}(\sigma, L)$ is smaller than $c_{in}(k, \sigma, L)$. We conducted analogous experiments with linearly increasing cost of intervention, varying the maximum possible cost (at the end of the case), which confirmed that the patterns described above remain the same. To answer RQ3, we have empirically confirmed that the alarm system achieves a benefit as

discussed in Section 3.3 in case the cost of the undesired outcome is sufficiently higher than the cost of the intervention and/or the cost of the intervention is sufficiently higher than the cost of compensation.

6   Related Work

The problem of cost-sensitive training of machine learning models has received significant attention. For example, Elkan [5] analyzes the notion of misclassification cost and defines conditions under which a misclassification cost matrix is reasonable. Turney [13] examines a broader range of costs in the context of inductive concept learning. This latter study introduces the notion of cost of intervention, which we include in our proposed cost model. These approaches, however, do not take into account the specific costs that arise in prescriptive process monitoring.

Predictive and prescriptive process monitoring are related to Early Classification of Time Series (ECTS), which aims at classifying a (partial) time series as early as possible, while achieving high classification accuracy [14]. To the best of our knowledge, works [15,16,17] are the only ECTS methods trying to balance accuracy-related and earliness-related costs. However, these approaches assume that predicting a positive class early has the same effect on the cost function as predicting a negative class early, which is not the case in typical business process monitoring scenarios, where earliness matters only when an undesired outcome is predicted.

Works [18,19] focus on alarm-based prescriptive process monitoring, but only allow alarms to be raised when a given state of the process is reached. This moment might potentially be late to mitigate the consequences, which would have been possible if the alarm was raised earlier. Furthermore, our approach does not require an explicit modelling of the process states. Last but not least, they rely on a fixed-threshold alarming mechanism provided by process owners, as opposed to our empirical thresholding approach. Gröger et al. [20] is an existing approach to provide recommendations, but it misses the two core elements of our proposed prescriptive process monitoring framework, i.e., cost models and earliness.

7   Limitations and Future Work

While the scenarios discussed in Boxes 1-3 show that the proposed framework is versatile enough to cover a variety of cases, the current version of the framework relies on two main assumptions. First, it assumes that an alarm always triggers an intervention, thus ignoring that a process worker might in some cases decide not to or be unable to intervene. Additionally, the current version of the framework considers each case in isolation, omitting the overall workload of the process workers, which in reality is an important factor for determining the number of alarms that can be acted upon. This limitation can be lifted by, e.g., combining the alarm system with [21], which proposes a recommender system that optimizes suggestions in case of concurrent process executions. A second

limitation of the framework is that only one possible type of intervention is envisaged. This assumption can be lifted by extending the framework so that the cost of an intervention can vary depending on the specific action suggested by a recommender system.

Next to these limitations, we acknowledge the importance of further investigation on the applicability of the framework in practice. In particular, in the future, we aim at collaborating with companies and institutions to study whether process stakeholders are able to define the costs in a natural and simple way. Also, we plan to further investigate the consequences of incorrect and/or imprecise instantiations of the cost models. Furthermore, the current evaluation is limited to measuring the benefit of the alarm system in an offline manner, while a more thorough evaluation would consist in deploying the alarming mechanism in a real organization and making an end-to-end comparison of the costs before and after the deployment of the alarm system. However, this is a difficult task for two main reasons. First, companies need to be willing to let the technique really influence the process executions. Second, the end-to-end effectiveness analysis cannot be conducted without coupling the alarm system with a recommender system: if the system raises proper alarms, but inappropriate interventions are taken, the system would still be ineffective. Another avenue for future work is to extend the framework with active learning methods in order to incrementally tune the alarming mechanism based on feedback about the relevance of the alarms and the effectiveness of the interventions.

8 Conclusion

This paper outlined an alarm-based prescriptive process monitoring framework that extends existing predictive process monitoring approaches with the concepts of alarms, interventions, compensations, and mitigation effects. The framework incorporates a cost model to analyze the tradeoffs between the cost of intervention, the benefit of mitigating or preventing undesired outcomes, and the cost of compensating for unnecessary interventions induced by false alarms. The cost model allows one to estimate the benefits of deploying a prescriptive process monitoring system for the purposes of return on investment analysis. Additionally, the framework incorporates a technique to optimize the alarm generation mechanism with respect to a given configuration of the cost model and a given event log. An empirical evaluation on real-life logs showed the benefits of applying this optimization versus a baseline where a fixed likelihood score threshold is used to generate alarms, as considered in previous work in the field.

References

1. Maggi, F.M., Di Francescomarino, C., Dumas, M., Ghidini, C.: Predictive monitoring of business processes. In: Proc. of CAiSE, Springer (2014) 457–472
2. Metzger, A., Leitner, P., Ivanovic, D., Schmieders, E., Franklin, R., Carro, M., Dustdar, S., Pohl, K.: Comparing and combining predictive business process monitoring techniques. IEEE Trans. Systems, Man, and Cybernetics: Systems **45**(2) (2015) 276–290

3. Teinemaa, I., Dumas, M., La Rosa, M., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. arXiv preprint arXiv:1707.06766 (2017)

4. Dees, M., de Leoni, M., Mannhardt, F.: Enhancing process models to improve business performance: A methodology and case studies. In: Proc. of CoopIS, Springer (2017) 232–251

5. Elkan, C.: The foundations of cost-sensitive learning. In: Proc. of IJCAI. Volume 17., Lawrence Erlbaum Associates Ltd (2001) 973–978

6. Sheng, V.S., Ling, C.X.: Thresholding for making classifiers cost-sensitive. In: AAAI. (2006) 476–481

7. Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Proc. of NIPS. (2011) 2546–2554

8. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems. JMLR **15**(1) (2014) 3133–3181

9. Olson, R.S., La Cava, W., Mustahsan, Z., Varik, A., Moore, J.H.: Data-driven advice for applying machine learning to bioinformatics problems. In: Proc. of Biocomputing, World Scientific (2017)

10. de Leoni, M., van der Aalst, W.M.P., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Information Systems **56** (2016) 235–257

11. Zadrozny, B., Elkan, C.: Learning and making decisions when costs and probabilities are both unknown. In: Proc. of KDD, ACM (2001) 204–213

12. Platt, J., et al.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Advances in large margin classifiers **10**(3) (1999) 61–74

13. Turney, P.D.: Types of cost in inductive concept learning. In: Proc. of the Cost-Sensitive Learning Workshop. (2002)

14. Xing, Z., Pei, J., Philip, S.Y.: Early classification on time series. KAIS **31**(1) (2012) 105–127

15. Mori, U., Mendiburu, A., Dasgupta, S., Lozano, J.A.: Early classification of time series by simultaneously optimizing the accuracy and earliness. IEEE Trans. Neural Netw. Learn. Syst. (2017)

16. Dachraoui, A., Bondu, A., Cornuéjols, A.: Early classification of time series as a non myopic sequential decision making problem. In: ECML PKDD, Springer (2015) 433–447

17. Tavenard, R., Malinowski, S.: Cost-aware early classification of time series. In: ECML PKDD, Springer (2016) 632–647

18. Metzger, A., Föcker, F.: Predictive business process monitoring considering reliability estimates. In: Proc. of CAiSE, Springer (2017) 445–460

19. Di Francescomarino, C., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring. IEEE Trans. Services Computing (2017)

20. Gröger, C., Schwarz, H., Mitschang, B.: Prescriptive analytics for recommendation-based business process optimization. In: Proc. of BIS, Springer (2014) 25–37

21. Conforti, R., de Leoni, M., La Rosa, M., van der Aalst, W.M.P., ter Hofstede, A.H.M.: A recommendation system for predicting risks across multiple business process instances. Decision Support Systems **69** (2015) 1–19