Estimating the Impact of Incidents on Process Delay

Felix Mannhardt
Technology Management
SINTEF Digital
Trondheim, Norway
Email: felix.mannhardt@sintef.no

Petter Arnesen
Mobility and Economics
SINTEF Building and Infrastructure
Trondheim, Norway
Email: petter.arnesen@sintef.no

Andreas D. Landmark
Technology Management
SINTEF Digital
Trondheim, Norway
Email: andreas.dypvik.landmark@sintef.no

***Abstract*—Process mining reveals how processes in organisations are actually performed and pinpoints deviations from the desired process execution. Process delay is one type of deviation that can be detected. Specific activities may take longer than expected or the waiting times between activities may deviate from service agreements. However, the quantification of processing or waiting times is often only the starting point in identifying the underlying root causes for process delay. One such root cause are adverse incidents in the environment of the process such as malfunctioning of supporting systems or unavailability of resources. Data about these external factors is often neither included in the event log nor recorded precisely enough to be directly linkable to a specific set of process instances. This paper presents a method for estimating process delay caused by incidents for which only the approximate occurrence time is known. We link incidents that are recorded in an incident log to process delay and calculate the effect of incidents on process delay using a Markov chain Monte Carlo sampling (MCMC) approach. Our proposed method was evaluated in a project conducted with the infrastructure manager of the Norwegian railway system. We applied it to a large event log of more than 120 million events capturing block-level movements of trains in the railway network and estimated the impact on process delay of about 50 000 infrastructure-related incidents. This showed that the method is useful for providing decision support and insights on the effects of maintenance. Since then the method has become part of the standard toolbox of the infrastructure manager.**

I. Introduction

While a process can be viewed as a sequence of events, with timings and ordering, that can be investigated with process mining methods [1]; in most processes it is also fruitful to discuss factors that negatively impact the process execution. These are often colloquially referred to as adverse events. These events are not necessarily a part of the process, but occurrences that directly or indirectly affect activities of a process. Statistical process control generally distinguishes between common-cause variations and special-cause variations [2], which correspond well to internal and external factors.

Common-cause variations, often internal to the process, are events caused by phenomena constantly active within a system, they can be probabilistically predicted, or simply seen as the natural variation or noise within a system. Typical common-cause variations are variable process performance due to capacity problems or difference process variants. Special-cause variations, often external to the system, are unpredictable and, even when internally sourced, variations outside the historical evidence base. Concrete examples are power outages, extreme weather conditions, etc. The latter source of delay often has an indirect causal connection with the execution of process activities. Therefore, the attribution of delay (or establishing a more direct cause-and-effect relation) in which the special-cause variation is isolated from the common-cause variation is often tricky.

We address the problem of estimating the impact of incidents on the performance of a process (special-cause variations) based on an event log, as well as a separate, non-integrated, incident log. The event log stores the process execution in terms of activity sequences and their timestamps as usually assumed in process mining. The incident log is a supplementary data source storing information on the occurrence of incidents with a possible influence on the process. Such incident logs, however, are likely not to contain precise information on the exact boundaries of an incident's influence on the process. Often, incidents are manually recorded after the fact or the exact time boundaries of their influence on the process are unknown.

Our contribution is an estimation method for process delay caused by external incidents. First, our method quantifies process delay based on comparing process performance to the typical performance as observed before and after the incident in an event log. Then, it connects this delay to an incident based on a Markov chain Monte Carlo (MCMC) estimation of the time window in which the incident influenced the process performance. The proposed method was evaluated in depth using a large event log obtained from railway traffic in the Norwegian railway network and corresponding incidents from a maintenance management system. In this scenario, estimation of process delay is necessary as the maintenance system contains only unreliable information on the exact occurrence time frame of incidents making it impossible to directly link the incidents to delay in the process.

This remainder of the paper is structured as follows. Section II expands on our motivation to study the delay estimation problem and illustrates it using a railway network process scenario; Section III presents the process delay estimation method; and Section IV describes the evaluation. Finally, Section VI concludes with an outlook on future work.

II. Motivation

Figure 1 illustrates the process delay estimation challenge. Often the execution of a *process* is logged in an *event log*. This

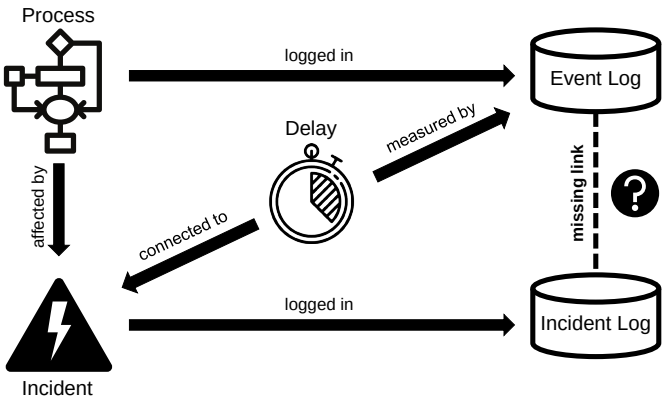

Fig. 1. Incidents affect the process execution and may cause process delay, which can be measured from the event log of a process. Incidents are also logged in an incident log, but cannot be directly linked to the events due to imprecise recording of the exact boundaries of an incident.

TABLE I
AN EXAMPLE RAILWAY TRAFFIC CONTROL EVENT LOG TAKEN FROM THE NORWEGIAN RAILWAY NETWORK WHICH IS USED AS INPUT TO OUR METHOD. TABLE ADAPTED FROM [3].

id	train	time	schedule	station	track	type
e_1	407	06:30:43	06:30:00	OPD	-	arrival
e_2	407	06:47:52	06:45:00	OPD	OPD-FGH	departure
e_3	407	06:53:27	06:51:30	FGH	OPD-FGH	arrival
e_4	407	06:53:49	06:52:00	FGH	FGH-UBG	departure
e_5	407	07:02:49	07:00:30	UBG	FGH-UBG	arrival
e_6	407	07:03:25	07:01:00	UBG	UBG-BAK	departure
e_7	407	07:11:19	07:08:00	BAK	UBG-BAK	arrival
e_8	407	07:13:15	07:09:00	BAK	-	departure
e_9	42	09:30:03	09:29:00	BAK	-	arrival
e_{10}	42	09:31:54	09:30:00	BAK	BAK-UBG	departure
...	...	...	...	...	...	

TABLE II
AN EXAMPLE INCIDENT LOG TAKEN FROM THE NORWEGIAN RAILWAY NETWORK WHICH IS USED AS INPUT TO OUR METHOD.

id	object	location	time
i_1	EH-MAS-123	km 453	2017-12-04 06:58:00
i_2	SA-DRV-123	km 442	2017-12-06 13:23:00
i_3	SA-SIK-123	km 320	2017-12-12 20:51:00
...	...	...	...

allows us to measure the performance of individual process instances or even specific process activities. By comparing to the normal process performance or to service agreements it is possible to measure process *delay*. Condition monitoring data provides information on adverse *incidents* that may affect the process performance. However, one challenge with such *incident logs* is often a weak or non-existent causal relation to operational performance metrics of the supported process. Establishing a causal relationship between the condition or state of physical infrastructure and the performance in use is impossible without perfect information. The goal of our work is to estimate this missing link between event log and incident log and connect individual incidents to the observed process delay.

We illustrate the problem of process delay estimation using an example taken from the railway domain in which adverse incidents on the railway infrastructure delay the block level movements of trains throughout the railway network. Railways are, from a statistical process control perspective, a process only barely in control. Schedules is an idealistic plan, and performance will usually be worse that planned. Hence, running with delays is the common case. This makes it extra difficult to determine and attribute *additional* lapses in performance (e.g. delays) over the common performance on the instance level. Common sources for process delays are passenger-related, operator-related, and infrastructure-related delays. These also interact through multiple complex pathways. So, the perfect separation of causality is assumed impossible. Moreover, the source of delay are interdependent, with several paths of correlation between them, necessitating an estimation-based approach to delay-allocation. Note that the estimation method is not limited to physical infrastructure as adverse incidents may also affect resources required in more traditional business processes such as loan applications or call centre handling.

Looking at the railway scenario from a process mining perspective, each run of a train through the network is a process instance and the block-level movement between stations can be modelled as process activities. In most railway networks, train movements are recorded in railway traffic control logs such as the one in Table I. Traffic control logs contain the running time of trains between stations and the dwelling time on stations. Based on this data, an event log suitable for measuring process delay can be built [3], [4]. We consider the movement between stations (column *track*) as process activity, which results in a process as depicted in Figure 2. Activities represent the movement of trains between stations (or intermediate measurement point) and the time spent between activities (shown on the edges) is the dwelling time on a station. Then, we can compute the delay by comparing the actual running time with the scheduled running time or the average running time in the past days.

Next to traffic control logs, the railway Infrastructure Manager (IM) manages all infrastructure condition related events and work orders in a maintenance management system. From this system an incident log as shown in Table II can be extracted. Each incident relates to a certain *object* that can be associated to a specific track by its location. The time at which the incident occurred is recorded, but due to manual registration often it is only an approximation of the actual time. Specifically, the time boundaries in which the incident may have had a influence, i.e., from its occurrence until the incident was resolved, are not recorded or not available with sufficient quality. Given this data the IM wants to connect process delay (i.e. late trains) to specific incidents in order to make informed decision on maintenance investments.

III. DELAY ESTIMATION METHOD

The proposed process delay estimation method infers the missing link between occurrences of *adverse incidents*, for which the exact time boundaries are unknown, and *process*

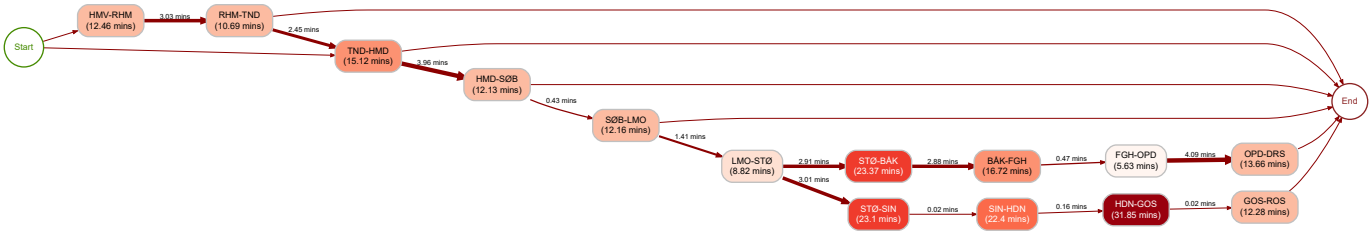

Fig. 2. Process map (directly-follows relations) of a small part of the Norwegian railway network discovered using half a year of traffic control logs for trains riding from Trondheim towards Oslo when using 90% of the most common traces. Note that on some stations trains do not stop, which explains the very low time waiting at some station (edges).

delays as measured from an event log. First, we describe the inputs and assumptions of our method. There are three inputs:

- an event log L of the process,
- a log I of adverse incidents;
- a relation $I2A$ connecting incidents to activities.

An event log L is defined over a set of unique events E and a set of activities A. It consists of traces $s \in L$ that are sequences of events, i.e., $L \subseteq \mathbb{P}(E^*)$ such that each event $e \in E$ occurs in one and only one trace. Each event $e \in s$ corresponds to the occurrence of some distinct activity from a pre-defined set of activities $act_E(e) \in A$ together with two timestamps: the start time of the activity $time^s_E(e) \in \mathbb{N}$ and the completion time of the activity $time^c_E(e) \in \mathbb{N}$. Such event log is also denoted as *activity log* as the two standard life-cycle transitions *start* and *complete* are combined in one event. Generally, non-compliant event logs can be pre-processed in an activity-log format even in the presence of noise, e.g., by using alignment-based conformance checking [6]. For convenience, we define $dur_E(e) \in \mathbb{N}$ with $dur_E(e) = time^c_E(e) - time^s_E(e)$ to return the total duration of an activity execution.

The set of potential adverse incidents I contains all those incidents which may have an impact on the process execution. For each incident $i \in I$ its time of occurrence: $time_I(i) \in \mathbb{N}$ is recorded. Note that this recording may have been done manually and, thus, could be not reliable, e.g., the incident may actually have occurred before this time.

The incident-activity relation $I2A \subseteq I \times A$ connects each incident to the activity most likely affected by the incident. We required this relation to avoid spurious delay registration and focus on activities for which we can assume a likely causal relation between the incident and the process performance. Obtaining $I2A$ is not always straightforward, but often it can be established automatically, e.g., in our railway scenario we can use available documentation to relate the location of objects for which an incident is registered to the track on which the train rides. Similarly, existing process models used to determine for which activities resource affected by an incident are required.

Taking this input, our method computes the delay attributable to each incident, i.e., $D \subseteq I \times \mathbb{Q}$. We determine the delay using the following four steps for each incident $i \in I$ and its set of related process activities $A_i = \{a \mid (i,a) \in I2A(i)\}$:

1) we estimate the normal performance of the process activity;
2) we classify process activity instances around the approximate time of incident into three classes: severely delayed, delayed, normal;
3) we use MCMC sampling to determine in which time frame and how likely the process performance was influenced by the incident;
4) we accumulate the difference between the normal process performance and the observed performance for all process instances executing the activity within the estimated time frame.

In the next four sections, we describe each step in detail for a single incident and its related process activity.

A. Estimating the normal process performance

First, we need to determine the normal performance of the process without influence of an incident. As our definition of process performance is relative to the activity which is considered to be connected to the incident through some causal link (required resource etc.), we primarily consider the local process performance in terms of the service time of that activity.

There are several ways to estimate what is considered normal. We determine the normal process performance based on the average performance in the time preceding and succeeding the incident. Our proposal does not make assumptions on Service Level Agreements (SLA) as we aimed to also identify the impact of incidents on process delay even when the delay is within a given SLA. This is important when the delay estimation is to be used for a process-improvement scenario since problems need to be identified before a SLA is violated.

Assume that incident i occurred at time $\mu_{t,i} = time_I(i)$. To determine the normal performance, we take the *average* and *standard deviation* of the process performance for each activity $a \in A_i$ observed in the time windows $[\mu_{t,i} - t_{N2}, \mu_{t,i} - t_{N1}]$ and $[\mu_{t,i} + t_{N1}, \mu_{t,i} + t_{N2}]$. Moreover, we exclude outliers with a very high or low duration. For example, in our use case, we excluded some unusually slow freight trains, which appeared regardless of incidents.

Given an event log E, its set of traces L, and the process activity $a \in A_i$ of interest. Let $E^a_{std} \subset E$ be the set of events

observed in the *normal state* time window:

$$E^a_{std} = \{e \in E \mid act_E(e) = a \wedge (time_E(e) \in [\mu_{t,i} - t_{N2}, \mu_{t,i} - t_{N1}] \vee time_E(e) \in [\mu_{t,i} + t_{N1}, \mu_{t,i} + t_{N2}])\}$$

We compute the average μ^a_{std} and standard deviation σ^a_{std} as usual:

$$\mu^a_{std} = \frac{\sum_{e \in E^a_{std}} (dur_E(e))}{|E^a_{std}|} \quad (1)$$

$$\sigma^a_{std} = \sqrt{\frac{\sum_{e \in E^a_{std}} (dur_E(e) - \mu^a_{std})^2}{|E^a_{std}| - 1}} \quad (2)$$

There are several considerations to be taken into account when choosing t_{N1} and t_{N2}. Parameter t_{N1}, the time distance from the assumed time of the incident, needs to be large enough so that the incident cannot possibly have had an influence. Similarly, t_{N2} should be large enough so that the effect of unrelated delays before or after the incident on the normal process performance estimate is negligible. For example, in our case we choose t_{N1} to be 24 hours and t_{N2} to be three days.

B. Classifying the duration of activity executions

Having determined the normal process performance in a standard situation, for each activity $a \in A_i$ we build a sorted sequence of events $seq^i_E(a) : A \rightarrow E^*$ from the executions of activity a, which are recorded in the event log in the time window $[\mu_{t,i} - t_{N1}, \mu_{t,i} + t_{N1}]$ directly around the suspected time of the incident:

$$seq^i_E(a) = \langle e_1, \ldots, e_n \rangle \; s.t. \; \forall_{1 \leq j < k \leq n} (\{e_j, e_k\} \subset E \wedge a = act_E(e_j) = act_E(e_k) \wedge t_j = time^c_E(e_j) \wedge t_k = time^c_E(e_k) \wedge t_j < t_k \wedge \{t_j, t_k\} \subset [\mu_{t,i} - t_{N1}, \mu_{t,i} + t_{N1}])$$

Hereafter, we simply refer to this sequence as $s_{a,i}$, i.e., $s_{a,i} = seq^i_E(a)$.

We classify each event $e \in s_{a,i}$ into one of three classes: $C = \{0, 1, 2\}$ using function $\gamma : E \times A \rightarrow C$ with:

$$\gamma(e, a) = \begin{cases} 1, & \text{if } dur_E(e) < 1.5\,\sigma^a_{normal} + \mu^a_{normal} \\ 2, & \text{otherwise} \\ 3, & \text{if } dur_E(e) > 3\,\sigma^a_{normal} + \mu^a_{normal} \end{cases}$$

Class 1 are **normal** instances in which the execution time is in the standard range for activity a that was observed before and after the incident. Class 2 are **delayed** instances which considerably longer. Class 3 are **severely delayed** activity executions with an unusually long execution time.

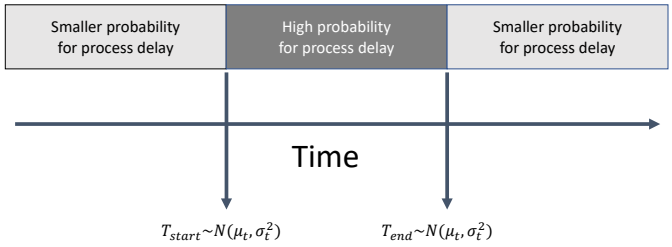

Fig. 3. We assume that the probability for process delay is increased in a time frame around the approximate time of the incident.

C. Estimating the influence of an incident

Now we have all the ingredients to estimate the influence on the execution time of process activities. As illustrated in Figure 3, we denote with t_{start} and t_{end} the time window in which incident i had an influence on activities A_i, which we want to estimate. We will use the notation $P(\cdot)$ for probability distributions, and assume apriori for each $a \in A_i$ that the probability distribution to register an event $e_j \in s_{a,i}$ from event log E at time $t_j = time^c_E(t)$ as being classified with class $\gamma(a, e_j) = c_j$ outside of the influence of the incident $t_j < T_{start}$ or $t_j > T_{end}$ has a discrete probability distribution $P_0(c_j|p_0)$. Conversely, the probability distribution for an activity execution in which $T_{start} < t_j < T_{end}$ is $P_1(c_j|p_1)$.

Parameters p_0 and p_1 are case dependant parameters that indicate the probabilities to observe the three different classes of delay. For example, in our specific case we choose the parameters as $p_0 = (0.94, 0.055, 0.005)$ and $p_1 = (0.93, 0.06, 0.01)$. In word, we assume apriori that there is a slightly higher probability to observe delayed or severely delayed events within the time boundaries in which the incident was influential on the system. These parameters were in our case tuned by running several pilot estimations and compare the results to cases annotated by experts from the Norwegian railway IM.

As an alternative to manual parameter tuning, these parameters could also be estimated by assigning a less informative prior distribution to the parameters p_j for $j = 0, 1$. For instance using a Dirichlet distribution as in [5]. However, simulating them along-side the other parameters in the posterior, would be much more computationally expensive, requiring all incidents to be considered in one long MCMC run. Therefore, in this first approach to solving this problem we propose tuning these parameter values base on problem experience and a smaller more know data. Or as another alternative p_j for $j = 0, 1$ could be estimated from a separate dataset including both incident-free or known incident cases. For other problems, directly assigning vaguer priors might also be a possibility that works well, however this approach was not investigated in this paper but will be subject to further research.

For each sequence s_a of events we assign a parameter $IO_a = 0, 1$ to denote whether or not the sequence of events related to activity a is not affected by the incident or affected by the incident, respectively. To this parameter we assign

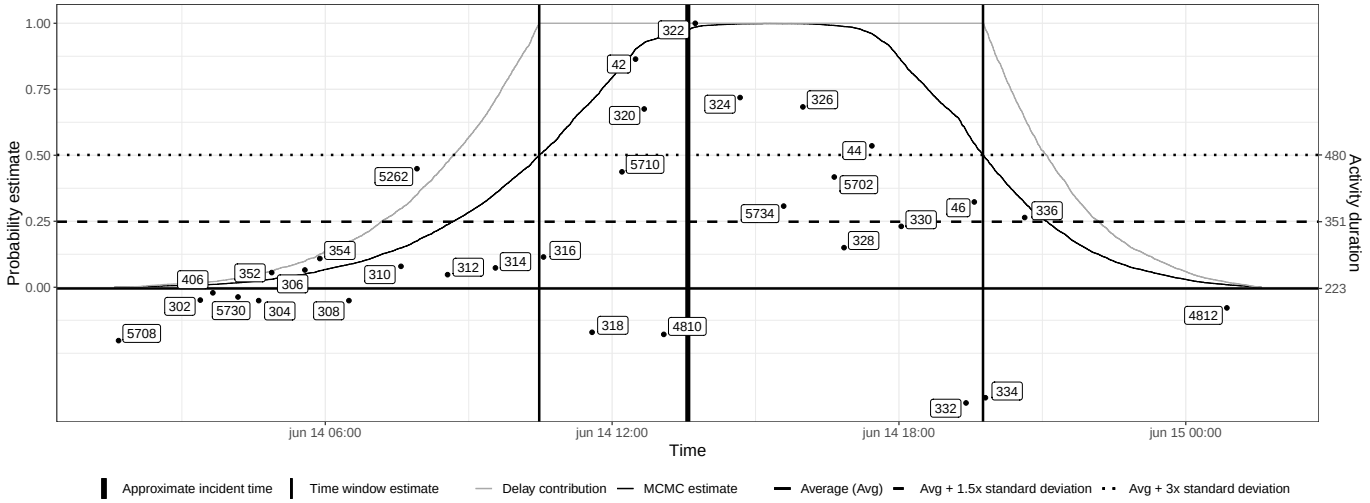

Fig. 4. Example of the estimation of the influence of an incident on activity durations of one specific activity. Each dot represents an activity execution, which is a train running along a certain track. The thick black vertical line depicts the assumed time at which the incident occurred. The horizontal lines show the estimates of the normal performance $\mu^a_{std}, \mu^a_{std} + 1.5\sigma^a_{std}$, and $\mu^a_{std} + 3\sigma^a_{std}$. Our simulation results in the black curve that shows the probability according to which a certain timestamp was placed between T_{start} and T_{end}. The grey line depicts the factor with which a prolonged activity duration contributes to the calculated process delay.

apriori the distribution $P(IO_a) = 0.5$ for $IO_a = 0, 1$. For the start and end times we assume apriori that $T_{start}, T_{end} \sim N(\mu_{t,i}, \sigma_t^2)$ with $T_{start} < T_{end}$, where $\mu_{t,i}$ is the assumed time of the incident and $\sigma_t = 4$ is a case dependant parameter. By Bayes theorem the posterior distribution for T_{start} and T_{end} is

$$P(T_{start}, T_{end}|E, A_i, IO, p_0, p_1, \mu_{t,i}, \sigma_t) \propto$$
$$I(T_{start} < T_{end}) P(T_{start}|\mu_{t,i}, \sigma_t) P(T_{end}|\mu_{t,i}, \sigma_t) \cdot$$
$$\prod_{a \in A_i} P(IO_a) \left(IO_j \prod_{e_j \in s_a} [g_1(j) + g_0(j)] \right),$$

where $I(\cdot)$ is the identity function being 1 if the argument is true and 0 otherwise, and

$$g_1(j) = I(t_j \in [T_{start}, T_{end}]) P_1(c_j|p_1),$$
$$g_0(j) = (1 - I(t_j \notin [T_{start}, T_{end}])) P_0(c_j|p_0).$$

In our use case we simulate 20000 iterations using the Metropolis-Hastings algorithms using multiple different proposal distributions for T_{start} and T_{end} and estimate the marginal aposterior distributions $P(T_{start}|E, A_i, IO, p_0, p_1, \mu_{t,i}, \sigma_t)$ and $P(T_{end}|E, A_i, IO, p_0, p_1, \mu_{t,i}, \sigma_t)$. This gives us an estimation of the most probable start and end time of the influence that the incident had on process delay.

D. Attributing process delay to an incident

Figure 4 gives an example of how the proposed estimation method works for a single activity when being applied to an event log in which activities instances are the trains driving between stations. It shows an excerpt of the registered activity durations obtained from the Norwegian railway IM for a single activity (i.e., single track) from several process instances (individual trains). Even though the registered time for the incident was around 13:30, there was already some significant delay before that time. The black curve indicates the probability according to which a certain timestamp was placed between T_{start} and T_{end}.

The delay for a single event e can be computed by comparing the recorded activity duration to the average duration in a normal situation, i.e., $delay_a(e) = dur_E(e) - \mu^a_{std}$. However, there are several ways to accumulate the individual delay to the total process delay attributable to the incident. A straightforward method is to multiply the delay recorded by event e with the probability estimate at time $time^c_E(e)$. We found in the validation of our case that this often lead to an under-approximation of delay. Also, in practice a delay is either caused by an incident or not caused by an incident. Therefore, directly using the simulation results can be problematic for interpretability of the results by process stakeholders. Thus, we adopted the following method.

We fully count the delay for every event for which the probability estimate is at least 50%, i.e., in at least half of the simulated samples $time^c_E(e)$ falls in the window $[T_{start}, T_{end}]$. This captures the intuition that in a core time frame the delay of each activity instance is fully attributed to the incident. For the remainder of the activity instances, we add only a fraction of the delay since the influence of the incident is less obvious. This yields the grey curve in Figure 4 that we obtain by scaling the simulation results with a factor of 2 and counting all delay caused by events that occur at a timestamp for which this scaled probability estimate exceeds 1.0 in full. For example, the delay of the process instance 46 is counted fully, whereas the delay for the process instance 336 is discounted with a factor of approximately 0.25.

In summary, we compute the total delay $d_E(i) \in \mathbb{Q}$ for

incident i connected to activities A_i:

$$d_E(i) = \sum_{a \in A_i} \sum_{e \in seq_E^i(a)} (delay_a(e) \cdot min(1, 2 \cdot p_e))$$

where

$$p_e = P(time_E^c(e) \in [T_{start}, T_{end}] | E, A_i, IO, p_0, p_1, \mu_{t,i}, \sigma_t)$$

based on the simulation.

We repeat the procedure and obtain the accumulated delay d_i for each incident i and build the set $D = \{(i, d) \mid i \in I \wedge d = d_E(i)\}$.

IV. Case: Norwegian railway traffic

We validated the proposed method using the process of train traffic in the Norwegian railway network. Our data source is a large event log from the traffic control system as well as the corresponding infrastructure incident log.

Norwegian railway infrastructure is owned and managed by a single Infrastructure Manager (IM). The IM is in charge of the infrastructure, including monitoring infrastructure condition, incident handling and preventative and corrective maintenance. The operation of rolling stock on the infrastructure is an open market, and currently there are 10 operators with a license for traffic on the general infrastructure. The majority of the infrastructure is a single-track infrastructure (94% single track, 65% electrified), in a star-shaped network around the capital. The topology and restrictions in single-track operation means that adverse incidents and their corrective maintenance is a important limiting factor for railway performance.

Infrastructure condition related events and work orders reside in a traditional maintenance management system, whilst the traffic data (i.e. system performance metrics) resides in an event log which automatically samples the track-side train detection in order to provide automatic second-resolution punctuality information. Prior to the algorithm there were no automatic allocation of delays, but only manual attribution. This attribution was conducted by dispatchers, acting on information conveyed by multiple sources (telephone, status logs, etc.). In a pre-study we identified that this manual attribution has obvious signs of batching, missing or obviously erroneous attribution.

A. Set-up

We implemented the proposed method in R and applied it to a large event log with more than 110 million events describing block-level movements of trains that was obtained from the traffic control system of the IM. Whilst being a centrally controlled system, the event log data was of varying quality which led to challenges similar to more traditional business process mining scenarios. For example, some of the tracks are still manually recorded. Such manual recording, data transfer issue, and non-standard train traffic caused only partially ordered events, swapped events, and several other issues. We used knowledge on the actual infrastructure, i.e., the de-jure process model, to mitigate such problems similar to conformance checking methods [6].

We obtained the incident-activity relation for 50 000 geo-located incidents that were recorded in the maintenance management system by using the location of the incident and lookup of the two closest stations to each side of the track. We excluded incidents that evidently cannot affect the performance of the process, e.g., incidents regarding the displays on a station.

For the tuning of parameters (p_0 and p_1), we observed that they are sensitive to large changes, however, small changes in parameters do not change the result significantly. Thus, the most important lesson when choosing these parameters is that an occurrence of a delay is only slightly more probable under an active incident, the relative difference here being more important than the actual level on the probabilities.

We repeated the estimation 20 times for each incident using the parameters indicated when presenting the method and compared the results to evaluate whether the MCMC simulation converged to a stable solution. Also, we investigating a large number of trace-plots for all simulated parameters, that showed convergence after the proposed number of iterations.

With repetitions, the computation took on average 6 minutes for a single incident including data retrieval on a single computation thread using standard server hardware. Note that the calculation can be trivially parallelised which brings the computation time down a level that is feasible in practice. The results were written back to the data warehouse environment of the IM for reporting and planning purposes.

B. Validation

In lack of a ground truth, we validated the results by comparing the results to a database of delays that were manually allocated to some incidents by train operators. Among the process instances for which manual registration data was available (only about 10% of the total data), we found that in most cases (74%) our approach found some or even all the delays manually allocated. Surprisingly, we found that in 36% of the cases, we did not find any of the delay attributed to incidents manually. We conducted an interview with a train operator to clarify the difference. Several possible reasons for this discrepancy were identified, which indicate the differences are mainly due to data quality issues.

The primary reason is that, due to manual nature of the delay registration and batch registration after a busy shift, sometimes trains are registered that have been already delayed before reaching the position of the incident (i.e., before executing the process activity in question). Moreover, we found that in some parts of the railway network the data quality of the traffic registration is poor, which results in missing delay as our proposal only takes the existing data into account. Another reason are knock-on effects of delayed trains to other trains in the railway network. These effects are not captured by this proposal. In practice, we already integrated the knock-on effect estimation approach presented in [7] for the single-track part of the network. However, we consider this as out of scope for this work.

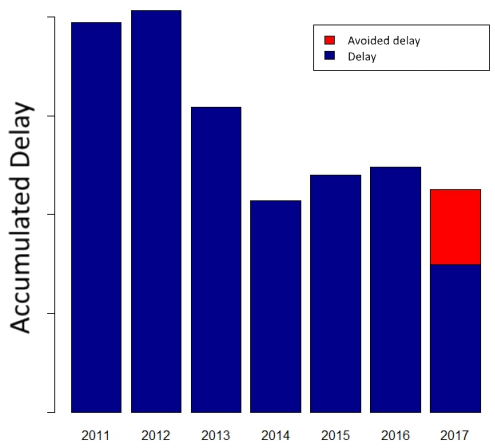

Fig. 5. Accumulated delay for a specific part of the Norwegian railway network as estimated based on the proposed method. The red bar in 2017 indicates the likely impact of incidents that were avoided by a preventive maintenance system. The y-axis has been concealed due to confidentiality.

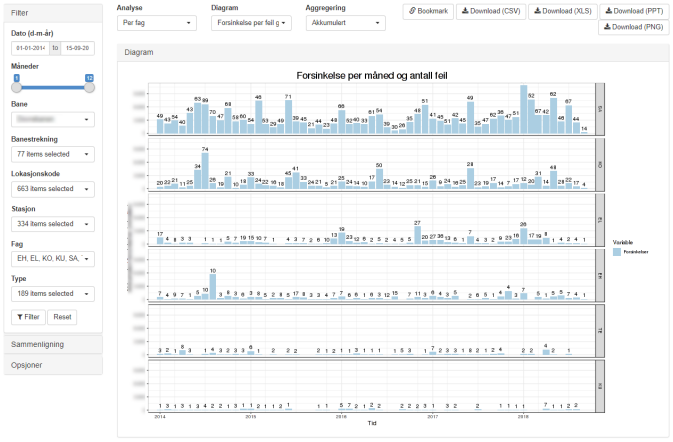

Fig. 6. The analysis dashboard built on top of the delay estimation data.

C. Results

There are many opportunities to use the process delay data produced by our method. We present two use cases for the data produced on the Norwegian railway traffic process.

In the first use case, we leverage the fact that it is possible to automatically obtain a large amount of delay data and that this data can be connected to a single resource in the infrastructure. In this way, it is possible to get a reliable estimate of the typical delay effect on the process that originates from a specific failing resource. Recently, the Norwegian IM has invested in preventive maintenance solutions for several heavily used turnouts. Upgraded turnouts raise an alarm when the equipment is likely to fail. We used data on these alarms, which can be seen as prevented incidents, and the aggregated median estimated process delay from the years beforehand to quantify the delay-reduction effect of this investment in predictive maintenance. The bar plot in Figure 5 shows the estimated savings in term of prevented delay. Such insights can be very valuable for evidence-based decision support.

In the the second use case, we used the unprecedented scale of the delay database, in comparison to manual registration, to open up new possibilities for exploratory data analysis and reporting. For this purpose, we built an analysis dashboard that can be used by the Norwegian IM. The dashboard has been used in several workshops and is in ongoing use to identify reoccurring issues with infrastructure and their effect on delays. A distinct advantage of this fully data-driven approach is that also the effect of many small impacts on process delay can be quantified. Whereas the reasons for failures resulting in rare large delays are well-known and can be manually investigated, the effect of many small incidents is difficult to establish but may well contribute more towards the overall customer satisfaction.

V. Related work

Investigating the performance of processes using event logs is one of the main tasks of process mining. A seminal work in this regard is [8] which establishes the notion of alignments to project reliable performance information extracted from event logs on process models. In [9] the observed performance of a process is captured with stochastic Petri nets. Decision trees are used in [10] to analyse influential factors of business process performance. Differently to our method, these factors are assumed to be internal to the process and, thus, already integrated with the process events. In [11] the correlation between resource workload and processing speed in a process is investigated based on an event log, however, no process delay is quantified.

Another type of methods aims to predict the process performance (e.g., remaining time, time to next activity). Like our setting the work in [12] considers not only intra-case features but takes a global view on all running cases. However, external incidents are not considered. In [13] the process prediction considers the influence of an environmental state, which is similar to our use of the incident log as external factor but with the aim to predict performance.

Besides these process-model-based approaches, there are also several proposals to investigate the process performance using visual analytics. A basic technique for comparing the performance across several traces is the Dotted Chart [14] in which events are visualised on a 2D scatter plot. Two more recent proposal are the ProcessProfiler3D [15], in which a general framework for 3D visualization of process performance is proposed, and the Performance Spectrum Miner [16], which focuses on visualizing performance variability over time. While useful for exploratory analysis of causes for process delay, it is not automatically attributing delay to incidents.

Regarding our case study, in railway research the effect of unplanned events such as infrastructure and rolling stock malfunctions, passenger-related incidents or and inclement weather is well known [17]. Interdependencies between in-

dividual delays and, so called, knock-on delays have been studied [7], [18]. Also, there is some research that took a process mining perspective on railway operations [3], [4], [19]. However, to the best of our knowledge, there is no work trying to quantify the effect of incidents on the overall delay of a process based on events logs.

VI. Conclusion

The process-oriented view on event data in process mining has led to a large number of methods that reveal insights about process execution. However, there is relatively little work on the impact that external factors have on processes even though processes are seldom executed in isolation, and events that are not directly related to the execution itself may have a large impact on it. Such *adverse incidents* are often also registered, but rarely in the same system, sometimes only manually, and often with considerable uncertainty regarding the exact time frame in which the incident had influence on the process. Thus, it is difficult to quantify their impact on the process execution.

The main contribution of our work is a method for process delay estimation caused by adverse incidents, which uses process event logs and some minimal approximate information on the incidents. We use Markov chain Monte Carlo (MCMC) sampling to determine the most likely times at which the influence of an incident on process performance started and ended. Then, we use the estimated likelihoods to accumulate the overall delay, i.e., difference in performance from a normal situation, that can be attributed to the incident. We implemented and evaluated the method using a large database of incidents and railway traffic control logs in the Norwegian railway system between 2011 and 2018. The evaluation showed that the method can reliably assign process delay to incidents and thta it is useful for showing the effect of investments in predictive maintenance. Finally, our method puts focus on the overall effect of many minor infrastructure failures individually resulting in small delays, which otherwise would be difficult to establish.

There are several options to improve the proposed method by addressing its limitations in future work.

First, we aim to apply the method to non-infrastructure related processes. In the railway case the incident-activity relation was easy to obtain, but in a general business context it may be more difficult to determine this relation.

Second, multiple co-occurring incidents that affect the same activity pose an issue for the proposed method both due to possibly multi-modal delay distributions as well as the difficulty to divide the delay properly. This, as well as, knock-on and complex queuing effects are limitations of our method and areas for future work. Regarding knock-on effects we did initial research based on the method presented in [7]; however, this method is not straightforward to apply it to the case of generic business processes without additional assumptions.

Last, the Posterior distribution which is simulated using MCMC uses a defined set of parameter values in the prior distribution. These parameter values do not generalize to other problems, and would need to be calibrated in each case. An alternative that could be explored is to define a less informative prior distribution, for instance using hyper distributions and simulating parameters p_0 and p_1 as well, however this comes with a high computational price.

Acknowledgment

This research was in part supported by Bane NOR.

References

[1] W. M. P. van der Aalst, *Process Mining - Data Science in Action, Second Edition*. Springer, 2016.

[2] W. E. Deming, *Out of the Crisis*. Cambridge University Press, 1986.

[3] F. Mannhardt and A. D. Landmark, "Mining railway traffic control logs," in *21st EURO Working Group on Transportation Meeting, EWGT 2018*, ser. Transportation Research Procedia, vol. 37, 2019, pp. 227–234.

[4] P. Kecman and R. M. P. Goverde, "Process mining of train describer event data and automatic conflict identification," in *Computers in Railways XIII*. WIT Press, Sep. 2012.

[5] P. Arnesen, T. Holsclaw, and P. Smyth, "Bayesian detection of change-points in finite-state markov chains for multiple sequences," *Technometrics*, vol. 58, no. 2, pp. 205–213, 2016.

[6] J. Carmona, B. van Dongen, A. Solti, and M. Weidlich, *Conformance Checking*. Springer International Publishing, 2018.

[7] A. Ø. Sørensen, A. D. Landmark, N. O. Olsson, and A. A. Seim, "Method of analysis for delay propagation in a single-track network," *Journal of Rail Transport Planning & Management*, vol. 7, no. 1-2, pp. 77–97, Jun. 2017.

[8] W. M. P. van der Aalst, A. Adriansyah, and B. F. van Dongen, "Replaying history on process models for conformance checking and performance analysis," *WIREs Data Min Knowl Discovery*, vol. 2, no. 2, pp. 182–192, 2012.

[9] A. Rogge-Solti, W. M. P. van der Aalst, and M. Weske, "Discovering stochastic petri nets with arbitrary delay distributions from event logs," in *BPM 2013 Workshops*, ser. Lecture Notes in Business Information Processing, vol. 171. Springer, 2014, pp. 15–27.

[10] B. Wetzstein, P. Leitner, F. Rosenberg, I. Brandic, S. Dustdar, and F. Leymann, "Monitoring and analyzing influential factors of business process performance," in *2009 IEEE International Enterprise Distributed Object Computing Conference*. IEEE, Sep. 2009.

[11] J. Nakatumba and W. M. P. van der Aalst, "Analyzing resource behavior using process mining," in *Business Process Management Workshops*. Springer Berlin Heidelberg, 2010, pp. 69–80.

[12] A. Senderovich, C. D. Francescomarino, C. Ghidini, K. Jorbina, and F. M. Maggi, "Intra and inter-case features in predictive process monitoring: A tale of two dimensions," in *BPM*, ser. Lecture Notes in Computer Science, vol. 10445. Springer, 2017, pp. 306–323.

[13] F. Folino, M. Guarascio, and L. Pontieri, "Discovering context-aware models for predicting business process performances," in *OTM Conferences (1)*, ser. Lecture Notes in Computer Science, vol. 7565. Springer, 2012, pp. 287–304.

[14] M. Song and W. M. van der Aalst, "Supporting process mining by showing events at a glance," in *Workshop on Information Technologies and Systems (WITS'07)*, 2007, pp. 139–145.

[15] M. Wynn, E. Poppe, J. Xu, A. ter Hofstede, R. Brown, A. Pini, and W. van der Aalst, "ProcessProfiler3d: A visualisation framework for log-based process performance comparison," *Decision Support Systems*, vol. 100, pp. 93–108, Aug. 2017.

[16] V. Denisov, D. Fahland, and W. M. P. van der Aalst, "Unbiased, fine-grained description of processes performance from event data," in *BPM 2018*, ser. Lecture Notes in Computer Science, vol. 11080. Springer, 2018, pp. 139–157.

[17] N. O. Olsson and H. Haugland, "Influencing factors on train punctuality — results from some norwegian studies," *Transport policy*, vol. 11, no. 4, pp. 387–397, 2004.

[18] H. Flier, R. Gelashvili, T. Graffagnino, and M. Nunkesser, "Mining railway delay dependencies in large-scale real-world delay data," in *Robust and Online Large-Scale Optimization*, ser. Lecture Notes in Computer Science. Springer, 2009, vol. 5868, pp. 354–368.

[19] G. Janssenswillen, B. Depaire, and S. Verboven, "Detecting train reroutings with process mining," *EURO Journal on Transportation and Logistics*, vol. 7, pp. 1–24, Apr. 2017.