A Principled Approach to Mining From Noisy Logs Using Heuristics Miner

Philip Weber, Behzad Bordbar and Peter Tiňo
School of Computer Science, University of Birmingham, UK
Email: {p.weber,b.bordbar,p.tino}@cs.bham.ac.uk

***Abstract*—Noise is a challenge for process mining algorithms, but there is no standard definition of noise nor accepted way to quantify it. This means it is not possible to mine with confidence from event logs which may not record the underlying process correctly. We discuss one way of thinking about noise in process mining. We consider mining from a 'noisy log' as learning a probability distribution over traces, representing the true process, from a log which is a sample from multiple distributions: the 'true' process model and one or more 'noise' models. We apply this using a probabilistic analysis of the Heuristics Miner algorithm, and demonstrate on a simple example. We show that for a given model it is possible to predict how much data is needed to mine the underlying model without the noise, and identify differences in the the robustness of Heuristics Miner to different types of noise.**

I. Introduction

Process mining is the learning of models of business processes, from event logs produced by the information systems used by the business. Process mining can be broadly split into three main areas [1]. We are concerned in this paper with process discovery, producing models to show the activities which take place and the relations between them.

We use as a running example the artificial process illustrated in figure 1. This is a simplified process for ordering a product from a supplier. When an order is received, stock is checked, and either the item picked from the warehouse, or the order rejected. Despatch and billing take place in parallel, then after checking payment, a receipt is issued or the payment chased before the order is closed. Abstracting from detail, the 'trace' of a single enactment of the process may be encoded as a string *iabdefgo*. This process can be considered as a probability distribution over such strings.

One key challenge in process mining (C6 in the Process Mining Manifesto [2]) is mining from noisy logs. In machine learning, noise generally refers to data errors such as signal error, variations in measurements, or random errors in data labels for classification. This would relate in process mining to problems in the recording of event logs. However, in process mining the term 'noise' tends to be used to refer to infrequent behaviour [1]. In either case we face the same problem. We wish to use some of the evidence in the event log to build our process model, while ignoring other evidence, and to end up with a model of 'reasonable complexity'.

To the best of our knowledge there is no standard or rigorous method for defining noise in the process mining context nor the effect which it has on the learning behaviour of algorithms.

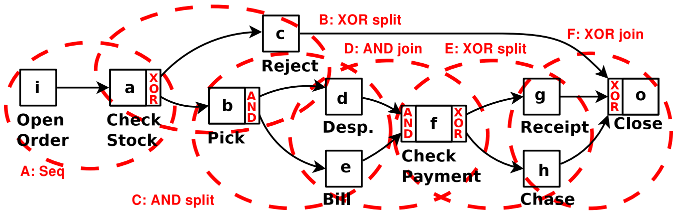

Fig. 1. Simplified Business Process for fulfilling an Order, highlighting Process Structures and types of Splits and Joins).

Without such a foundation, it is not possible to compare and predict the behaviour of algorithms in noisy situations. Practically, we cannot describe 'how much' noise a particular algorithm can handle, nor how much data we should use to mine the true underlying model but exclude noise. These are important questions, since errors such as disk failure, software bugs or erroneous use of systems can lead to errors in logs. Process mining is also often used in complex business environments such as healthcare (e.g. [3]), where to produce manageable process models of the core process behaviour, it may be desirable to leave out infrequent process paths.

In this paper we describe for the first time one formal model of noise in process mining. We present a summary of a probabilistic analysis of the Heuristics Miner algorithm [4], which is designed to handle noise. We apply this analysis to the running example (figure 1) and simple artificial models of noise affecting it. We conclude that using our model of noise and this type of probabilistic analysis, conclusions can be drawn about how much noise of different types a process mining algorithm can handle. We also show that Heuristics Miner is better able to deal with some types of noise than with others, and that for known models, we can use these methods to predict the number of traces needed to safely mine the underlying model without being affected by the noise.

II. Probabilistic View of Process Mining

To investigate a formal model of noise and its effect on learning process models, we outline a probabilistic view of process discovery, described in full in [5]. We view the control-flow aspect of business processes as distributions over strings of symbols which represent process traces. Activities are represented by symbols $\{a, b, \ldots\}$ from a finite alphabet Σ, traces as strings $x \in \Sigma^+$. A business process $\mathcal{M}$ is modelled by a probability distribution $P_{\mathcal{M}}$ over traces. The probability of trace x is $P_{\mathcal{M}}(x)$, such that $\sum_{x \in \Sigma^+} P_{\mathcal{M}}(x) = 1$. An event log (workflow log) $\mathcal{W}$ is a finite multiset over Σ^+, drawn *i.i.d.* from $P_{\mathcal{M}}$. The basic task of a process discovery algorithm is to learn from $\mathcal{W}$ a distribution $P_{\mathcal{M}'}$, to approximate $P_{\mathcal{M}}$ (which may be unknown). We consider here only acyclic process models (no repeating activities), and place restrictions on processes equivalent to those used elsewhere, e.g. [6]: a process has a single start activity i and end activity o; the events of activities' occurrence are atomic (take no time) and are recorded in $\mathcal{W}$ as they occur. We assume that the underlying process is unchanging while $\mathcal{W}$ is recorded.

We can represent these distributions using probabilistic automata (PDFA). Figure 2 represents the same process as figure 1, with the addition of transition probabilities. For example, the probability that after seeing ia in a trace, the next activity is c, is 0.1, and the probability of trace $iaco$ is $1.0 \times 1.0 \times 0.1 \times 1.0 = 0.1$. Automata are not convenient as a visual representation of more complex processes, since every state is explicitly represented, but are useful for us to illustrate the distributions represented by the underlying processes.

We write $\pi_{ab,\mathcal{M}}$ for the probability under $P_{\mathcal{M}}$ of ab occurring in a trace, and $\pi_{\rightarrow a}$ for the probability of 'reaching' a in the model. As we deal only with acyclic models, ab can only occur in any trace a maximum of once. We denote by $|ab|$ the number of times string ab occurs in the log. We will describe various process structures: $a \rightarrow b$ refers to the arc representing causal dependency between a and b; $a \rightarrow (b_1 \parallel b_2 \ldots)$ a split from activity a to parallel paths starting with $b_1, b_2 \ldots$; and $a \rightarrow (b_1 \# b_2 \ldots)$ a split from activity a to alternative paths starting with $b_1, b_2 \ldots$.

$\pi_n(E)$ is the probability of event E in an event log of n traces, e.g. $\pi_n(|ab| = x)$. $P_{HM,n}(S)$ is the probability of Heuristics Miner mining process structure S correctly from a log of n traces. DM_{ab} represents the Heuristic Miner Dependency Measure (DM) between a and b.

III. A Model of Noise in Process Mining

Noise is defined in [1], [2] as 'outliers' or exceptional events, i.e. parts of the process which occur infrequently. It is assumed that the mined model should not include such behaviour, which would cause a visually cluttered or 'spaghetti' model, in which the main process behaviour is not clear. This differs from the standard machine learning view in which noise refers to erroneous data which occurs according to some model of noise. In the context of process discovery we would understand this as incorrect logging of events or their order. The reasoning behind the current process mining view is that since an algorithm cannot distinguish incorrect logging from exceptional events, true noise (data errors) should be cleaned from the log using expert input, prior to process mining, so that the mining algorithm can assume that the log reflects what really happened.

Similarly, Fahland *et al.* [7] consider that a log may be partitioned into three sets of traces, (i) those supported by an external model, (ii) those not supported because the model is incorrect, and (iii) those not supported because they represent

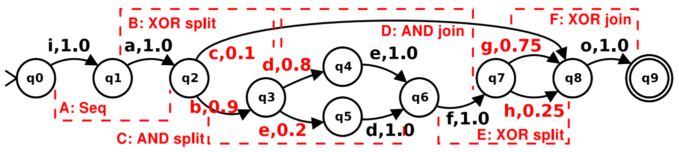

Fig. 2. Probabilistic Automaton (PDFA) equivalent of Figure 1. This is model $\mathcal{M}$, representing Distribution $P_{\mathcal{M}}$.

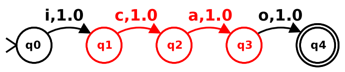

Fig. 3. Simple Noise Model $\mathcal{O}_1$, in which Activities a and c are swapped.

'noise'. An expert, or prior knowledge, is required to identify this partitioning and remove the noisy traces. The cleaned log is then used to 'repair' a pre-existing model.

The Heuristics Miner literature [4], [8] takes a more standard view of noise, although it does not consider a model of noise generation. Noise is described as external influences on the event log which cause five types of errors: deletion of the head, tail or part of the body of a trace, removal of one event, or interchange of events in a trace.

We describe a more formal view of noise in process mining, which covers both logging errors and infrequent behaviour. We consider traces in event log $\mathcal{W}$ to be drawn from two underlying probability distributions. Let $P_{\mathcal{M}}$ be a distribution over traces representing the true business process ('ground truth'), and $P_{\mathcal{O}}$ a distribution over all possible 'noise' traces. Event log $\mathcal{W}$ is then a sample from a mixture distribution $P_{\mathcal{W}}$ which is a combination of $P_{\mathcal{M}}$ and $P_{\mathcal{O}}$. Any trace $t \in \mathcal{W}$ is drawn with some fixed probability $0 < \kappa \ll 1$ from $P_{\mathcal{O}}$ (a 'noisy' trace), otherwise from $P_{\mathcal{M}}$ (a 'true' trace):

$$P_{\mathcal{W}}(t) = (1 - \kappa) P_{\mathcal{M}}(t) + \kappa P_{\mathcal{O}}(t). \tag{1}$$

We would like our process mining algorithm to output a model that supports $P_{\mathcal{M}}$, rather than any convolution of $P_{\mathcal{M}}$ and $P_{\mathcal{O}}$.

To illustrate, consider two ways in which noise might affect the running example. Firstly, the process may be followed incorrectly, for example activities executed or recorded out of prescribed order. If an order is rejected before stock is checked, trace $icao$ will be recorded. This is represented by automaton $\mathcal{O}_1$ (figure 3). Secondly, systems failures might cause incorrect recording of traces. For example, a low risk of disk failure during any activity gives a noise model supporting traces such as $iao, iabo, iabdo$ etc., illustrated by model $\mathcal{O}_2$, figure 4.

We have described just one possible view of noise, but having such a formal model allows investigation of a process mining algorithm to answer questions about how much and what types of noise the algorithm is robust to, and how noise in the log affects confidence in mining a correct process model. We next outline the main points of a probabilistic analysis of Heuristics Miner [4] (as implemented in ProM 5.2 [9]), which allows these questions to be answered for this algorithm. Due to space limitations we only give a summary here. A more complete analysis will be published elsewhere.

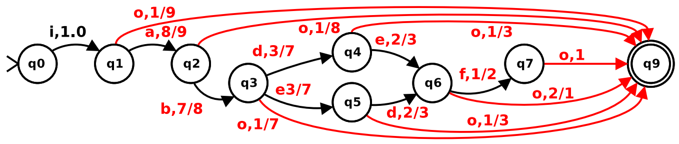

Fig. 4. Simple Noise Model $\mathcal{O}_2$ allowing any Activity to be followed by o.

IV. Overview of the Heuristics Miner Algorithm

The Heuristics Miner [4] (HM) is designed for mining from noisy logs. It uses frequencies of pairs of activities in the log to determine causal relations, splits and joins. It mines a 'Causal Matrix' (CM), visually represented (in ProM) as a 'Heuristic Net', a directed graph in which nodes represent activities and arcs represent causal dependencies. Splits and joins are separately annotated as representing exclusive choice (XOR) between subsequent activities, or to parallel paths (AND).

The model is constructed in three steps. First, a Dependency Matrix M is created. The elements of M are the values of the Dependency Measure (DM) (equation 2) between pairs of activities. Next, M is used to construct a Dependency Graph to describe the causal structure of the process. Finally, the types of splits and joins are determined as parallel (AND) or exclusive (XOR). (Here we ignore further steps to identify cycles and 'long-distance' relationships between activities).

HM uses threshold parameters, which with the DM give some control over the detail to include in the model. In this way noisy logs are handled by controlling the size and complexity of the mined model. The *'Use All-Activities-Connected Heuristic'* (UH) ensures the graph is connected: each activity other than the start and end activities has at least one successor and one predecessor. Creation of additional arcs is controlled by three parameters, *Positive Observations* (PO), *Dependency Threshold* (DT) and *Relative-to-Best* (RTB). To create an additional arc $a \rightarrow b$, substring ab must be seen in the log at least PO times, DM_{ab} must be larger than DT, and DM_{ab} must be within RTB of the DMs of the relations to existing successors of a or predecessors of b.

We next describe these steps for an underlying process $P_{\mathcal{M}}$. We assume the UH parameter is set to true.

A. Create Dependency Matrix

Let T be the set of activities recorded in $\mathcal{W}$ and produce a $|T| \times |T|$ Dependency Matrix M. Each element of M is the Dependency Measure (DM) between two activities, e.g.

$$\mathrm{DM}_{ab} = \frac{|ab| - |ba|}{|ab| + |ba| + 1}. \tag{2}$$

B. Create Dependency Graph

A node is created for each activity $t \in T$. A single start node is assumed: the activity for which none of the column entries in M are positive. In the same way, the single end node is the activity for which none of the row entries are positive. The graph is connected by identifying a successor and a predecessor activity for each remaining node. The successor for activity a is the activity corresponding to the largest DM in the row for a, and the predecessor for a is the activity corresponding to the largest DM in the column for a.

Additional arcs are created using the parameter settings. An arc $a \rightarrow b$ is created if

1) $\mathrm{DM}_{ab} > DT$,
2) $|ab| > PO$, and
3) $|\,\mathrm{DM}_{ab} - \mathrm{DM}_{ax}\,| < \mathrm{RTB}$ or $|\,\mathrm{DM}_{ab} - \mathrm{DM}_{yb}\,| < \mathrm{RTB}$,

where DM_{ax} is the next largest Dependency Measure in the row for a (successors of a), DM_{yb} the next largest in the column for b (predecessors of b).

C. Types of Splits and Joins

The main requirement for correct mining is for the dependency measures to be ordered correctly, to create the correct causal links in the Dependency Graph. An 'AND metric' is involved in deciding if splits and joins are AND or XOR, but we do not consider this as it does not affect the types of noise which we consider. We first consider Heuristics Miner mining from a noise-free log.

Consider an event log $\mathcal{W}$ of n traces, and DM_{ia} between two activities i and a which can only occur in one order, i.e. $\pi_{ia} \in [0,1], \pi_{ai} = 0$. DM_{ia} is a discrete random variable which tends to 1 as n increases. Now consider activities a and b which may occur in either order (interpreted as occurring in parallel). Without loss of generality let $0 < \pi_{ab} < \pi_{ba} < 1$. Then the expected value of DM_{ba} from this log converges to some value $d \in [0,1]$. For such $\mathrm{DM}_{ia}, \mathrm{DM}_{ba}$, since the log is randomly generated according to the underlying model, for low n DM_{ia} may be either less than or greater than DM_{ba}. However as n increases, we expect DM_{ia} to 'overtake' DM_{ba}, i.e. there will always be some number of traces $n' \in [0, \infty)$ above which $\mathbb{E}_{\mathcal{W}}[\mathrm{DM}_{ia}] > \mathbb{E}_{\mathcal{W}}[\mathrm{DM}_{ba}]$.

We need to know probabilities such as $\pi_n(\mathrm{DM}_{ia} > \mathrm{DM}_{ba})$, Let $A = \mathrm{DM}_{ia}, B = \mathrm{DM}_{ba}$ be random variables following unknown distributions with density functions g_A, g_B, CDF G_A, G_B and joint density $g_{AB}(A,B)$. Then

$$\pi_n(A > B) = \int_{A=-\infty}^{A=+\infty} \int_{B=-\infty}^{B=A} g_{AB}(A,B)\,dA\,dB. \tag{3}$$

Since business processes tend to be structured, it is not necessary for DMs to be correctly ordered across the whole matrix. In the next section we consider process structures (sequences of activities, XOR or AND splits and joins), and the correct ordering of the DMs involved in these structures.

V. Probabilistic Analysis of Heuristics Miner

To determine probabilities of one Dependency Measure (DM) exceeding another, we need to understand the probability distribution that the DM (2) follows. Since we assume no cycles, any activity pair ab occurs zero or one times in any trace, so the number of times ab occurs in event log $\mathcal{W}$ of n traces, is described by a Binomial distribution with probability parameter π_{ab}. The DM therefore follows a distribution which is the ratio of two 'Binomial-like' distributions, which are discrete and in general multi-modal and difficult to handle analytically. If we assume that the distributions can be approximated by Gaussians, then the theory in [10] can be used to derive formulae which approximate the DM density and cumulative distribution. The formulae are still difficult to work with analytically, especially since DMs cannot be assumed to be independent of each other, but numerical approximations can be used to obtain the probabilities of interest.

Business processes tend to be well structured [11]. We relate such structures to our probabilistic framework in [5]. In the next subsections we outline methods for obtaining the probability of Heuristics Miner correctly mining basic acyclic process structures from noise-free logs ($\kappa = 0$).

A. Sequences

If activities a and b form a sequence in the model then if a occurs, it is always immediately followed by b. In the simplest case that no other activity x can occur in parallel with a or b, neither ax nor xb is possible in the log. So if least one trace in the log contains ab, DM_{ab} will be the only positive DM in the row for a and the column for b in the Dependency Matrix. No traces will include ba. The UH parameter will ensure arc $a \to b$ is created, regardless of the the other parameters. The probability of discovery of a sequence is thus the complement of the probability that every trace in $\mathcal{W}$ will not contain ab,

$$\pi_{HM,n}(a \to b) = 1 - (1 - \pi_{ab})^n. \tag{4}$$

B. XOR Splits and Joins

An m-way XOR split occurs where there is a choice between m mutually exclusive paths through the model after activity a, each path starting with an activity $t_1, \ldots, t_m$. Similarly to discovery of a sequence, at least one trace in the log must contain ab_1, ab_2. and so on. With the same restrictions we may assume all $\pi_{b_i a} = 0$ and none of the activities b_i will occur together in a trace, so all $\pi_{b_i b_j} = 0$. Each b_i can only have a as a predecessor, and thus the PO, RTB and DT parameters are again not involved. Therefore

$$\begin{aligned} P_{HM,n}\big(a \to (b_1 \# \ldots \# b_m)\big) = \\ = 1 - \sum_{1 \le i \le m} (1 - \pi_{ab_i})^n + \sum_{1 \le i < j \le m} (1 - \pi_{ab_i} - \pi_{ab_j})^n - \\ \ldots + (-1)^m \big(1 - \sum_{1 \le i \le m} \pi_{ab_i}\big)^n. \end{aligned} \tag{5}$$

Equation (5) uses the 'inclusion-exclusion' principle for calculating the probability of a union of overlapping events. We assume successful mining of the split, reduce by the sum of the probabilities of the events that any activity pair ab_i is missing from the log. This double counts the probability of the events that any *pair of pairs*, e.g. both ab_i and ab_j, is missing. The third term adds this probability back in, and so on.

XOR joins are treated in the same way.

C. 2-Way Parallel Splits ('AND2')

A two way parallel split occurs where two paths through the model proceed in parallel, following activity i. Let one path start with activity a, the other with b. The simplest case is where no other parts of the model occur in parallel, i.e. $\pi_{ia} + \pi_{ib} = \pi_{\to i}$, and the parallel paths contain no other activities after a and b. Then note that $\pi_{ab} = \pi_{ia}, \pi_{ba} = \pi_{ib}$. There is only one free probability parameter; knowing π_{ia}, we know the other probabilities. Knowing either DM_{ia} or DM_{ba} also fully determines the other, and always $\mathrm{DM}_{ba} = -\mathrm{DM}_{ab}$. Without loss of generality we assume $\pi_{ib} > \pi_{ia}$, so we expect DM_{ab} to be negative and we consider only the requirements to ensure that with high probability $\mathrm{DM}_{ia} > \mathrm{DM}_{ba}$.

To mine the parallel split correctly, we require

1) $\mathrm{DM}_{ia} > \mathrm{DM}_{ba}$, otherwise b will be chosen instead of i as the predecessor of a and we have a sequence.
2) Either $\mathrm{DM}_{ia} > \mathrm{DM}_{ba} + \mathrm{RTB}$, $|ba| < \mathrm{PO}$ or $\mathrm{DM}_{ba} < DT$. This ensures the conditions in subsection IV-B are not met and the extra arc $b \to a$ will not be retained.
3) $|ia| > \mathrm{PO}$ and $|ib| > \mathrm{PO}$, else split will be XOR.

To calculate the probability of 'achieving' the first requirement, we need to integrate the joint distribution (3). Since $|ia|$ and $|ba|$ are fully correlated, the joint distribution of $\mathrm{DM}_{ia}, \mathrm{DM}_{ba}$ for a given number of traces n lies on a line, a curve as shown in figure 5. Thus we can marginalise one of the DM variables without loss of information, effectively projecting the joint distribution onto either axis.

Let $\mathrm{DM}_{ia,n} = \mathrm{DM}_{ba,n}$ be the dependency measures calculated for some n, defined in terms of unknown $p = \pi_{ia}$ for which the expected values of the DMs are equal, then

$$\mathbb{E}_{\mathcal{M}}[\mathrm{DM}_{ia,n}] = \mathbb{E}_{\mathcal{M}}[\mathrm{DM}_{ba,n}] \Rightarrow \frac{np}{np+1} = \frac{n\pi_{\to i} - 2np}{n\pi_{\to i} + 1},$$

for which there will be only one valid solution for p. Hence we obtain the value $\mathrm{DM}_{ia,n} = \mathrm{DM}_{ba,n}$, from this p. Since we are interested in the part of the distribution for which $\mathrm{DM}_{ia} > \mathrm{DM}_{ba}$, we integrate the PDF for DM_{ia} for $\mathrm{DM}_{ia} > \mathrm{DM}_{ia,n}$:

$$\pi_n(\mathrm{DM}_{ia} > \mathrm{DM}_{ba}) = \int_{\mathrm{DM}_{ia} = \mathrm{DM}_{ia,n}}^{\mathrm{DM}_{ia} = \infty} g(\mathrm{DM}_{ia}) d\,\mathrm{DM}_{ia}\,. \tag{6}$$

where $g(\mathrm{DM}_{ia})$ is the density function (PDF) of DM_{ia}.

To ensure the $b \to a$ arc is not retained (conditions in subsection IV-B), $\pi_n(\mathrm{DM}_{ia} > \mathrm{DM}_{ba} + \mathrm{RTB})$ is calculated by the same method, $\pi_n(\mathrm{DM}_{ba} < \mathrm{DT})$ by integrating a single DM distribution, and $\pi_n(|ba| < PO)$ from the Binomial distribution $|ba| \sim Bin(\pi_{ib}, n)$.

Finally, to correctly mine the 2-way AND split, both $|ia|$ and $|ib|$ must exceed PO. Here we must take account of other parts of the model: a trace may contain ia, ib, or neither, so $|ia|, |ib|$ are drawn from a Multinomial distribution with three outcomes with probabilities $\pi_{ia}, \pi_{ib}, 1 - \pi_{\to i}$. The cumulative Multinomial distribution can be used to calculate the probability of ia and ib occurring more than PO times.

Combining the previous requirements, the probability of

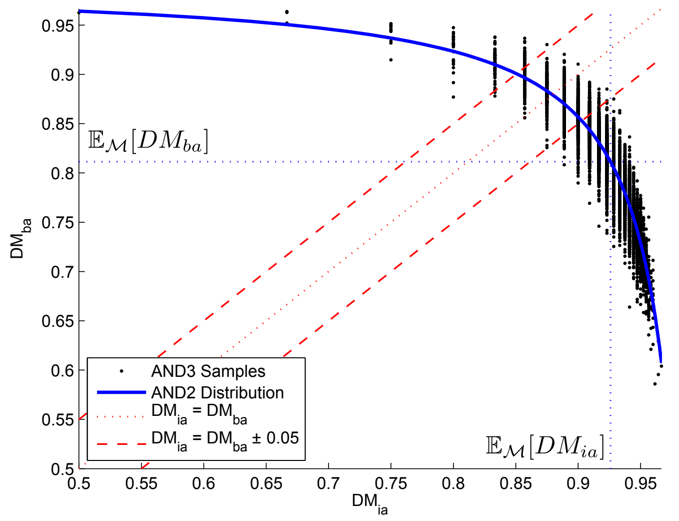

Fig. 5. Example of the Joint Distribution of $\mathrm{DM}_{ia}, \mathrm{DM}_{ba}$ in Three-Way Parallel split ('AND3'). The equivalent 'AND2' Distribution is also shown.

mining the split correctly is given by

$$\begin{aligned} P_{HM,n}\big(i \to (a \parallel b)\big) &= \pi_n(\mathrm{DM}_{ie} > \mathrm{DM}_{fe}) \\ &\times \pi_n\Big(|fe| < \mathrm{PO} \vee \mathrm{DM}_{fe} < \mathrm{DT} \\ &\qquad \vee (|\,\mathrm{DM}_{ie} - \mathrm{DM}_{fe}\,| < \mathrm{RTB}\Big) \\ &\times \pi_n\big(|ia| > \mathrm{PO} \wedge |ib| > \mathrm{PO}\big), \end{aligned} \tag{7}$$

where $e, f \in \{a, b\}, e \neq f$ such that $\mathbb{E}_{\mathcal{M}}[\mathrm{DM}_{ef}] > 0$.

D. 3-Way Parallel Splits ('AND3')

For a 3-way parallel split from i to paths beginning with a, b or c, there are 6 possible sequences of activities, $iabc, iacb, ibac, \ldots$, and therefore 3 pairs of Dependency Measure requirements, e.g. $\mathrm{DM}_{ia} > \mathrm{DM}_{ba}$ or $\mathrm{DM}_{ib} > \mathrm{DM}_{ab}$, etc. One of the pair is 'given' because either DM_{ab} or DM_{ba} will be negative, and the same for the other pairs. For the requirement $\mathrm{DM}_{ia} > \mathrm{DM}_{ba}$, the two DMs are no longer fully correlated but the correlation can be shown to still be negative. Figure 5 shows an example of the joint distribution. Effectively DM_{ba} varies, for given DM_{ia}, around the 'AND2' line described in the previous section. We can use the same method as a reasonable approximation to $\pi_n(\mathrm{DM}_{ia} > \mathrm{DM}_{ba})$.

E. Other Structures

Splits to more than three paths, complex splits and joins, other parts of the model which can occur in parallel, and parallel paths containing more than one activity, can be treated with extensions of these methods. Space does not permit discussion here. Instead, in the next section we investigate the effect of noise on the structures already discussed.

VI. Effect of Noise

Recall that we consider the event log $\mathcal{W}$ as a sample from a mixture distribution. Each trace is drawn either (with probability κ) from $P_{\mathcal{O}}$, a distribution over traces, representing a noise model, or (probability $1-\kappa$) from the true model $P_{\mathcal{M}}$.

We define the set of traces allowed by the true process model by $T_{\mathcal{M}}$, and 'unexpected' (noise) traces by $T_{\mathcal{O}}$. These sets are disjoint because by definition any trace supported by the true model is not noise, and any noise trace is not supported by the true model. The set of traces in the log, $T_{\mathcal{W}}$, is a subset of the union of $T_{\mathcal{M}}$ and $T_{\mathcal{O}}$ since the log is only a sample, and may include not all the traces supported by the models,

Since HM deals with pairs of activities, we also define three sets of pairs of activities, the possible pairs which may occur in traces from these sets. $B_{\mathcal{M}}$ contains all pairs of activities which have non-zero probability under $P_{\mathcal{M}}$, i.e. $B_{\mathcal{M}} = \{ab | \pi_{ab,\mathcal{M}} > 0\}$. Likewise $B_{\mathcal{O}}$ and $B_{\mathcal{W}}$. Again, $B_{\mathcal{W}}$ is a subset of $B_{\mathcal{M}} \cup B_{\mathcal{O}}$ but activity pairs may be in both $B_{\mathcal{M}}$ and $B_{\mathcal{O}}$ since the same pairs of activities may exist in traces from both the true and the noise models. The probability of an activity pair in the event log $\mathcal{W}$, is the weighted sum of its probability in the true and noise models,

$$\pi_{ab,\mathcal{W}} = (1-\kappa)\pi_{ab,\mathcal{M}} + \kappa\pi_{ab,\mathcal{O}}. \tag{8}$$

A model mined from event log $\mathcal{W}$ is at risk of two types of problem, described in the next subsections.

A. Extra XOR Splits and Joins

Recall that each trace begins with the same activity i, and ends with the same activity o. Then any 'noise' trace from $T_{\mathcal{O}}$ will partially match at least one true trace from $T_{\mathcal{M}}$. For any 'noise' trace, the first part (prefix, at least i) will match the prefix of a true trace up to some activity a after which they diverge. Likewise, the suffix will match the suffix of a true trace, from some common activity b to the end of the traces.

Therefore we have the risk that HM will create an extra XOR split $a \to (a' \# a'')$ at the divergence point, or an extra XOR join $(b' \# b'') \to b$ at the convergence point. Activities a', a'' are the activities following a in the traces from $T_{\mathcal{M}}$ and $T_{\mathcal{O}}$ respectively. Similarly b' and b'' are the activities preceding b. There may be multiple such splits and joins introduced by a trace which matches in multiple places.

Consider the risk of creating an unwanted XOR split. Let the true process model $\mathcal{M}$ contain a sequence $i \to a \to a'$, and the noise model introduces a pair $aa'' \in B_{\mathcal{O}}$. A new arc aa'' will be created in the model mined from $\mathcal{W}$ if the requirements in subsection IV-B are met, i.e.

$$\begin{aligned} |aa''| > PO \wedge \mathrm{DM}_{aa''} > DT \\ \wedge \Big(|\,\mathrm{DM}_{aa''} - \mathrm{DM}_{aa'}\,| < \mathrm{RTB} \\ \vee |\,\mathrm{DM}_{aa''} - \mathrm{DM}_{ea''}\,| < \mathrm{RTB}\Big), \end{aligned} \tag{9}$$

where e is an existing predecessor activity of a''. Therefore we can use at most n' traces for mining such that the probability of each one of these events is below some small $0 < \epsilon \ll 1$ representing an acceptable risk of the mined model being disrupted by noise.

Compare this with mining from noise-free logs, where we need at least n traces for confidence $1-\epsilon$ in mining the correct model. This suggests that for mining from noisy logs, there will be a range of traces within which mining will succeed.

B. Introduction of parallelism

If the pairs of activities $B_{\mathcal{O}}$ supported by the noise model include a pair (e.g. ba) that is the reverse of a pair (ab) from $B_{\mathcal{M}}$, then HM may conclude these are in parallel. If $P_{\mathcal{M}}$ contains sequence $i \rightarrow a \rightarrow b$, then if in $P_{\mathcal{O}}$ the reverse substring has non-zero probability, it also has non-zero probability in the event log. It is now possible to meet some of the requirements for mining a parallel split $i \rightarrow (a \parallel b)$. There are three ways in which this can happen,

1) arc $i \rightarrow b$ created because $\mathrm{DM}_{ib} > \mathrm{DM}_{ab}$,
2) arc $i \rightarrow b$ created because PO and DT requirements are met for ib, and DM_{ib} is within RTB of ab, or
3) arc $i \rightarrow b$ created because PO and DT requirements are met for ib, and DM_{ib} is within RTB of ia.

For safe mining all of these probabilities must be low.

C. Effect of noise on true probabilities

Since the event log $\mathcal{W}$ is a sample from a mixture of $P_{\mathcal{M}}$ and $P_{\mathcal{O}}$, noise may reduce the probabilities of seeing pairs of activities in the log, i.e. for all pairs of activities in $B_{\mathcal{W}}$, $\pi_{ab,\mathcal{W}} \leq \pi_{ab,\mathcal{M}}$. This is because some of the traces in $T_{\mathcal{O}}$ may not include some pairs of activities in $B_{\mathcal{M}}$, reducing the probability of including them in $\mathcal{W}$. Since these probabilities affect the probabilities of correct mining of structures, with noise it is likely that more traces will be needed for correct mining of process structures and thus of the full model.

VII. Experimental Evaluation

We used the methods described above to predict the number of traces needed for correct mining of the running example:

$$P_{HM,n}(\mathcal{M}) = P_{HM,n}(A) \times P_{HM,n}(B|A) \\ \times P_{HM,n}(C|B) \times P_{HM,n}(D|C) \times \ldots$$

where $P_{HM,n}(B|A)$ is the probability of correct mining of structure B given that we have successfully mined A[1]. The first row of table I shows the predicted numbers of traces needed for mining from noise-free logs, for various values of the HM parameters. The only parameter that has any effect with this model is PO, indicating that seeing enough traces to decide the split is parallel, not XOR, is the determining requirement.

We next applied the methods from the previous section to predict the effect on the running example of the two noise models outlined in section III. Noise model $\mathcal{O}_1$ (figure 3) allows activities a and c to be swapped, introducing the single risk that $a \parallel c$ would be mined instead of $a \rightarrow c$. The second model, O_2, allows for traces being truncated after any activity, with each 'noisy' trace having equal probability. New activity pairs $B_{\mathcal{O}} = \{io, ao, bo, do, eo, fo\}$ are introduced, risking new arcs $i \rightarrow o, a \rightarrow o$, etc., in the mined process model.

[1]In summary, we use the local substring probabilities in structure B rather than the global probabilities, and only consider traces that include B [5].

noise	defaults	RTB : 0.01	0.1	PO : 5	1	DT : 0.5	0.95
0	84	84	84	**49**	**45**	84	84
0.01	85	85	85	**49**	**45**	85	85
0.05	89	90	90	**52**	**47**	90	90
0.1	94	95	95	**55**	**50**	95	95
0.5	171	171	171	**100**	**95**	171	171
0.01	84	84	84	**52**	**52**	84	84
0.05	81	81	81	**48**	**48**	81	81
0.1	78	78	78	**45**	**44**	78	78
0.5	59	59	59	59	59	59	59

TABLE I
Predicted Numbers of Traces needed for correct mining with varying Amounts of Noise from $\mathcal{O}_1$ (Top) or $\mathcal{O}_2$ (Bottom).

We calculated the effect of various levels of noise by varying κ in the mixture model (1). The top half of table I shows the effect of κ on the number of traces to with high probability successfully mine the correct model from an event log $\mathcal{W}$ sampled from a mixture of $P_{\mathcal{M}}$ and $P_{\mathcal{O}_1}$. Success means mining a model which supports the traces in $T_{\mathcal{M}}$ but not those in $T_{\mathcal{O}_1}$. Again, only varying the PO parameter has any effect on the number of traces since having enough traces in the log to be confident of meeting the PO requirement for HM to decide the split is AND, rather than XOR, is the determining factor. The other parameters have no effect here since they affect when the $d \rightarrow e$ arc will be removed, but the split will still be XOR. Increasing noise increases the traces needed because the probability of the true traces is reduced, making PO traces harder to achieve.

Repeating for model O_2 (lower half of table I) we find counter-intuitively that increasing noise reduces the number of traces needed, until large amounts of noise are introduced. This is because traces from $\mathcal{O}_2$ include the pair be, required for the AND split, with higher probability than in traces from $P_{\mathcal{M}}$. The noise traces increase $\pi_{be,\mathcal{W}}$, increasing the likelihood of correct mining of the AND split. Eventually the number of traces reduces to the point where different structures are the limiting factor.

Table II shows predictions for the numbers of traces at which the different types of noise will with probability greater than 0.05 affect the mined model, i.e. above which mining becomes 'unsafe'. The top half of the table shows the predictions for the parallel structure $a \parallel c$ from $\mathcal{O}_1$. As we would expect, increasing noise reduces the safe number of traces for this risk. For the lowest noise, this happens when DM_{ic} increases to within RTB of DM_{ac}, which causes arc $i \rightarrow c$ to be created, as PO and DT are also achieved. This risk can thus be reduced by decreasing RTB or increasing DT. With more noise, the limiting factor is $\mathrm{DM}_{ic} > \mathrm{DM}_{ac}$, which is not affected by varying parameters.

Similarly, the lower half of table II shows the maximum numbers of traces safe to use for mining from event logs sampled from a mixture of $P_{\mathcal{M}}$ and P_{O_2}. Since there are six possible noise structures, and mining of any one of them represents a failure of mining, we recorded the minimum

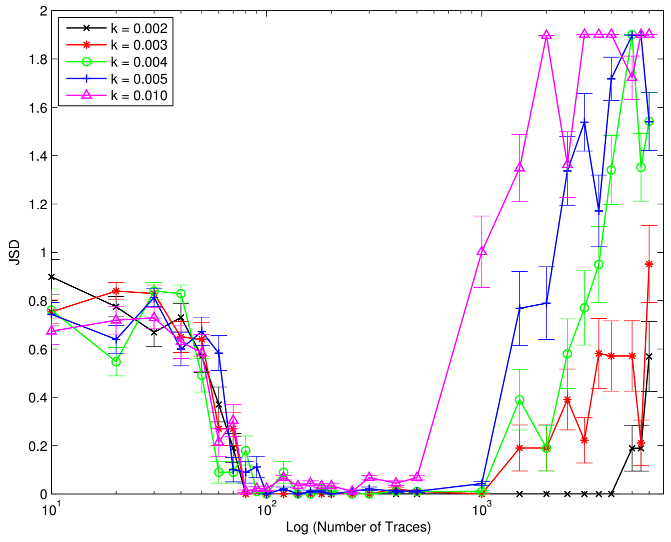

Fig. 6. Probability of Approximately Correct Model, mining from Logs from $\mathcal{M}$ mixed with $\mathcal{O}_1$, for Various Amounts of Noise (κ), HM Default Parameters ($\mathrm{PO} = 10, \mathrm{RTB} = 0.05, \mathrm{DT} = 0.9$).

number of traces at which the probability of any one noise structure exceeded 0.05. For this model, the limiting factor is mining arcs from d, e or f to o, since these occur in more noise traces. For low noise, this is limited by the probability of DM_{fo} being within RTB of DM_{co}, and the risk can be reduced by reducing RTB. For more noise, $|fo| > PO$ is the limiting factor, so reducing PO reduces the risk. When this is done, $\mathrm{DM}_{eo} > DT$ becomes the limit and DT can be used to control the risk of noise discovery. DT is effective in all cases.

We verified these predictions experimentally To simulate workflow logs from $\mathcal{M}$ affected by noise from $\mathcal{O}_1$, either $\mathcal{M}$ or $\mathcal{O}_1$ was chosen randomly according to the value of κ, then randomly walked to generate a single trace. This was repeated to generate sets of 10 logs in the MXML format, of various sizes from 10 to 6000 traces. A large 'ground truth' log of 10000 traces was also simulated from $\mathcal{M}$. The Heuristics Miner implementation and conversion plugins in ProM 5.2 [9] were used to mine process models from these logs and convert them to Petri nets. These Petri nets were converted to probabilistic automata by labelling their reachability graphs with maximum likelihood probability estimates derived from frequencies of activity pairs in the 'ground truth' log. A symmetrised Kullback-Leibler divergence (JSD distance) was calculated between the distributions represented by the ground truth and mined models, using methods from [12]. The distance was averaged for the 10 models mined from each log size and used to determine convergence to the true process.

Figure 6 shows the average distance between the ground truth and mined model, mining with from logs generated from the ground truth mixed with noise model $\mathcal{O}_1$, for various values of κ. The JSD distance is plotted against number of traces (log scale). For these experiments, we used the default HM parameters, $\mathrm{DT} = 0.9, \mathrm{RTB} = 0.05, \mathrm{PO} = 10$. The results confirm the concept of a range of traces within which

noise	defaults	RTB : 0.01	PO : 1	DT : 0.95
0.001	6676	**18308**	6676	**13058**
0.002	2714	**4158**	**2620**	**5025**
0.003	1559	1559	1559	1559
0.004	554	554	554	554
0.005	444	444	444	444
0.01	5622	**36700**	5622	**5878**
0.05	889	**6372**	889	**1178**
0.1	246	**2466**	**237**	**590**
0.3	84	84	**80**	**199**
0.5	51	**75**	**49**	**121**

TABLE II
PREDICTED NUMBERS OF TRACES FOR INCLUSION IN THE MINED MODEL OF NOISE FROM $\mathcal{O}_1$ (TOP) OR $\mathcal{O}_2$ (BOTTOM).

with high probability a correct model will be mined. The points of convergence to and divergence from correct mining are approximately in line with predictions.

VIII. Discussion and Related Work

The results show Heuristics Miner to be quite robust to the type of noise that risks additional XOR splits and joins, but less so to noise introducing parallelism. This is a general result, since the former can be controlled using the HM parameters, at the risk of omitting true low-probability arcs, but the latter will always affect the model once the 'noisy' Dependency Measure grows bigger than the true one. This is not affected by the parameter settings, unless the UH parameter is unset.

Our model of noise is general. We used only a single noise distribution, but the event log could contain traces from a mixture of multiple models. Noisy traces could come for example from models representing systems failure, process problems such as omission of audit checks, or rare scenarios of no interest. The effect on mining would be the same; noisy traces modifying the probabilities of activity pairs, and introducing new pairs, changing the probability of correct mining, and risking noisy structures affecting the mined model.

While we looked at one specific version of one algorithm, and only at discovery of process control-flow, the probabilistic interpretation of process mining is also general. Any process mining activity that uses an event log is learning from a random sample from underlying aspects of the true process, whether sequences of atomic activities, activity duration, or other data associated with processes or events. Only probabilistic statements can therefore be made about the correctness of results. Any mining algorithm can also be analysed probabilistically, in terms of how it uses the data in the log and the rules it applies to construct a process model.

Intuitively we might expect that with noisy data, using more data would give us a better model. For example, if estimating a measurement subject to some error, averaging repeat measurements will give a more accurate result. Process mining literature talks about the completeness of logs affecting the ability of algorithms to mine, e.g. whether every possible trace or pair of activities is in the log [1], [2]. However we have shown here that in the presence of noise, using more data is not sufficient to ensure correct mining. The process mining task is more complex than estimating a measurement, as are the algorithms involved. Completeness itself is a problem when the data is noisy. Instead we need to understand the affect of noise on the probabilities, and thus the likely composition of the log, and the behaviour of the mining algorithm.

Practically, while the Heuristics Miner parameters can be tuned interactively to control the visual complexity of a mined model, without an understanding of the model to be mined and the noise that may affect it, we can not be confident in the mining accuracy. In addition, there are complex interactions between the parameters, and without an understanding of how they interact with the particular model and the probabilities in it, the parameters cannot be objectively set. Depending on which requirement for mining a structure is hardest to achieve, tuning of different parameters may be most effective.

Our method provides one way of determining effective parameter settings. This question has previously been investigated for Heuristics Miner^{++} [13]. A method is presented in [15] to efficiently search the parameter space by factorising it to reduce the size of the search problem. This has the advantage of being applicable to any process, perhaps unknown, but even when factorised the search space may be large. The objective quality of the best model found is unknown since it is only compared with the log and with other possible models; no assessment is made of the probability of its correctness. Our method assumes a known model but makes probabilistic guarantees on the accuracy of the mined model. However, it requires significant initial analysis, although it should be possible to automate this. It also assumes structured processes and would need modification to deal with unstructured processes.

We plan to apply our analysis to other algorithms to understand better what factors affect different algorithms' behaviours in various situations. Elsewhere [5] we presented a detailed analysis of the Alpha algorithm [6], but Alpha cannot handle noise. It would be more interesting to apply our analysis to algorithms such as, Heuristics Miner^{++} [13] which extends Heuristics Miner to allow for activities with duration; the Flexible Heuristics Miner (FHM) [8] which enhances HM to improve mining of cyclic processes and describe splits and joins more flexibly; and Fuzzy Miner [14] which uses multiple notions of significance and correlation to mine complex and noisy logs. The complexity of the mined model is controlled using thresholds to remove or aggregate nodes and edges.

A limitation of our method is that the process of analysing an algorithm is difficult and needs to be applied from scratch to each algorithm. Our analysis of Heuristics Miner showed that a relatively simple algorithm can mask complex probabilistic behaviour. Analysis of more complex algorithms may be impractical. Future work should instead identify more general approaches to understand the behaviour of particular types of algorithms, and to use key characteristics of a process model to predict the amount of data needed for mining.

While it is useful to understand how much data is needed for known models, much process mining takes place from logs where the underlying model is not known. We plan within our framework to develop methods for understanding the probability of correctness of a mined model in such a situation.

IX. Conclusions

We considered for the first time a formal model of noise in process mining. Using a probabilistic understanding of process mining, and analysis of the Heuristics Miner algorithm, we demonstrated a method to determine the appropriate number of traces to use for mining a correct model, without mining the noise. This is of practical benefit, providing a method to be confident in mining a correct process model, showing that Heuristics Miner is more robust to certain types of noise, and providing insight into how to set the algorithm's parameters.

The full probabilistic analysis will be published elsewhere.

Acknowledgment

Philip Weber is supported by a Doctoral Training Grant funded by EPSRC and the School of Computer Science, University of Birmingham.

References

[1] W. M. P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.

[2] W. M. P. van der Aalst, et al. Process Mining Manifesto. In F. Daniel, K. Barkaoui, and S. Dustdar (eds.), *BPM Workshops (1)*, *LNBIP* vol. 99, pp. 169–194. Springer, 2011.

[3] R. S. Mans, M. H. Schonenberg, M. Song, W. M. P. van der Aalst, and P. J. M. Bakker. Application of Process Mining in Healthcare — a Case Study in a Dutch Hospital. In A. Fred, J. Filipe, and H. Gamboa, (eds.), *BIOSTEC 2008*, vol. CCIS 25, pp. 425–438, 2008.

[4] A. J. M. M. Weijters, W. M. P. van der Aalst, and A. K. Alves de Medeiros. Process Mining with the HeuristicsMiner Algorithm. *BETA Working Paper Series 166*, pages 1–34, 2006.

[5] P. Weber, B. Bordbar, and P. Tiňo. A Framework for the Analysis of Process Mining Algorithms. *IEEE Trans. Syst. Man Cybern. A, Syst. Humans (USA)*, PP(99):1 –15, 2012. accepted,

[6] W. M. P. van der Aalst, T. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Trans. Knowl. Data Eng.*, 16(9):1128–1142, 2004.

[7] D. Fahland and W. M. P. van der Aalst. Repairing Process Models to Reflect Reality. In A. P. Barros, A. Gal, and E. Kindler, (eds.), *BPM*, LNCS vol. 7481, pp. 229–245. Springer, 2012.

[8] A. J. M. M. Weijters and J. T. S. Ribeiro. Flexible Heuristics Miner (FHM). In *CIDM*, pp. 310–317. IEEE, 2011.

[9] B. F. van Dongen, A. K. A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and W. M. P. van der Aalst. The ProM Framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, (eds.), *ICATPN*, LNCS vol. 3536, pp. 444–454. Springer, 2005.

[10] G. Marsaglia. Ratios of Normal Variables. *Journal of Statistical Software*, 16(4):1–10, May 2006.

[11] W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.

[12] C. Cortes, M. Mohri, A. Rastogi, and M. D. Riley. Efficient Computation of the Relative Entropy of Probabilistic Automata. In J. R. Correa, A. Hevia, and M. A. Kiwi, (eds.), *LATIN*, LNCS vol. 3887, pp. 323–336. Springer, 2006.

[13] A. Burattin and A. Sperduti. Heuristics Miner for Time Intervals. In *Proceedings of ESANN 2010*, Bruges, Belgium, 2010.

[14] C. W. Günther and W. M. P. van der Aalst. Fuzzy Mining - Adaptive Process Simplification based on Multi-Perspective Metrics. In G. Alonso, P. Dadam, and M. Rosemann, (eds.), *BPM*, LNCS vol. 4714, pp. 328–343. Springer, 2007.

[15] A. Burattin and A. Sperduti. Automatic Determination of Parameters' Values for Heuristics Miner++. In *IEEE Congress on Evolutionary Computation*, pp. 1–8. IEEE, 2010.