Repairing Event Logs Using Timed Process Models

Andreas Rogge-Solti[1], Ronny S. Mans[2], Wil M.P. van der Aalst[2], and Mathias Weske[1]

[1] Hasso Plattner Institute at the University of Potsdam
Prof.-Dr.-Helmert-Strasse 2-3, 14482 Potsdam
{andreas.rogge-solti,mathias.weske}@hpi.uni-potsdam.de
[2] Department of Information Systems, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{r.s.mans,w.m.p.v.d.aalst}@tue.nl

Abstract. Process mining aims to infer meaningful insights from process-related data and attracted the attention of practitioners, tool-vendors, and researchers in recent years. Traditionally, event logs are assumed to describe the as-is situation. But this is not necessarily the case in environments where logging may be compromised due to manual logging. For example, hospital staff may need to manually enter information regarding the patient's treatment. As a result, events or timestamps may be missing or incorrect.

In this work, we make use of process knowledge captured in process models, and provide a method to repair missing events in the logs. This way, we facilitate analysis of incomplete logs. We realize the repair by combining stochastic Petri nets, alignments, and Bayesian networks.

Keywords: process mining, missing data, stochastic Petri nets, Bayesian networks.

1 Introduction

Many information systems record detailed information concerning the processes they support. Typically, the start and completion of process activities together with related context data, e.g., actors and resources, are recorded. In business process management, such event data can be gathered into logs. Subsequently, these logs can be analyzed to gain insights into the *performance* of a process. In many cases, information systems do not force the process participants to perform tasks according to rigid paths, as specified by process models. Rather, the process participants are responsible to track their manual work which is sometimes not reflected in the system. In other words, the event logs might be *incomplete* or noisy [1]. These data quality issues affect process mining methods and often lead to unsatisfactory results.

Existing approaches can be used to *repair* the model based on event data. However, if steps are recorded manually this may lead to misleading results as little weight is given to a priori domain knowledge. Therefore, we adopt a stochastic approach to modeling process behavior and introduce a novel approach to *repair event logs* according to a given stochastically enriched process model [2]. To model the as-is process we use Petri nets enhanced with stochastic timing information and path probabilities.

Y.T. Demey and H. Panetto (Eds.): OTM 2013 Workshops, LNCS 8186, pp. 705–708, 2013.

In fact, we use a variant of the well-known Generalized Stochastic Petri nets (GSPNs) defined in [3]. As a first step, using path probabilities, it is determined which are the most likely missing events. Next, Bayesian networks [4] capturing both initial beliefs of the as-is process and real observations are used to compute the most likely timestamp for each inserted entry. The complete procedure is described in more detail and evaluated in the technical report [5].

2 Realization of Repairing Logs

For this realization, we make the following assumptions:

- The supported models, i.e., the SPN models, are *sound*,and *free-choice*, but do not necessarily need to be (block-)structured. This class of models captures a fairly large class of process models and does not impose unnecessary constraints.
- The stochastic Petri net model is normative, i.e., it reflects the as-is processes in structural, behavioral and time dimension.
- Activity durations are independent and have normal probability distributions, containing most of their probability mass in the positive domain.
- The recorded timestamps in the event logs are correct.
- Each trace in the log has at least one event, and all events contain a timestamp.
- The activity durations of a case do not depend on other cases, i.e., we do not look at the resource perspective and there is no queuing.
- We assume that data is *missing at random* (MAR), i.e., that the probability that an event is missing from the log does not depend on the time values of the missing events.

The algorithm is depicted in Fig. 1, and repairs an event log as follows.

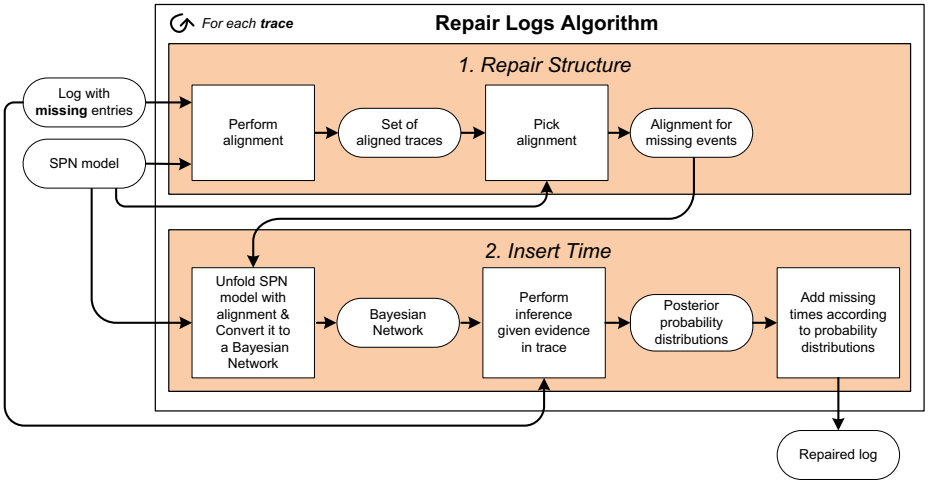

Fig. 1. The repair approach described in more detail

2.1 Repairing the Structure

For each trace, we start by repairing the structure. This becomes trivial, once we identified a path in the model that fits our observations in the trace best. The notion of cost-based alignments [6] is used for this part. We obtain a set of possible alignment candidates that are all cost-minimal in terms of costs for asynchronous moves.

In the next step, cf. box *Pick alignment* in Fig. 1, we decide which of the returned cost-minimal alignments to pick for repair. The algorithm replays the path taken through the model and multiplies the probabilities of the decisions made along the path. This allows us to take some probabilistic information into account, i.e., we can choose from the structural alignments one of the highest probability, or pick randomly according to the probability of such a path. Once we decided on the structure of how our repaired trace will look like, we can continue and insert the times of the missing events in the trace, i.e., the identified *model moves*.

2.2 Inserting Time

In the previous step, we identified the path through the SPN model. With the path given, we can eliminate choices from the model by removing branches in the process that were not taken. We unfold the net from the initial marking along the chosen path. Note that loops are but a special type of choices and will be eliminated from the model for any given trace. We transform the resulted unfolded model into a Bayesian network with a similar structure.

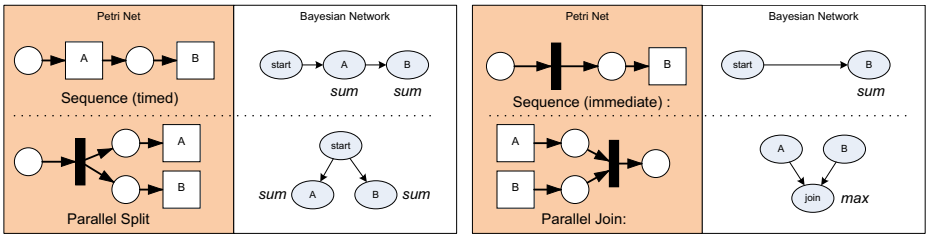

Fig. 2. Transformation of SPN models to Bayesian Networks

Fig. 2 shows the transformation of sequences, parallel splits, and synchronizing joins. These are the only constructs remaining in the unfolded form of the SPN model. In the resulting Bayesian network, we use the *sum* and *max* relations to define the random variables given their parents. More concretely, let t_i be a timed transition with a random variable with duration distribution $D_i(x)$ followed by another timed transition t_j with distribution $D_j(x)$ in a sequence. We can convert this fragment into a Bayesian network with random variables X_i and X_j. Then, the parent variable X_i has the unconditional probability distribution $\mathrm{P}(X_i \leq x) = D_i(x)$ and the child variable X_j has the conditional probability distribution $\mathrm{P}(X_j \leq x \mid X_i) = \mathrm{P}(X_j + X_i \leq x)$. For each possible value of the parent $x_i \in X_i$, the probability is defined as $\mathrm{P}(X_j \leq x \mid X_i = x_i) = \mathrm{P}(X_j + x_i \leq x) = D_j(x - x_i)$. This means that the distribution of X_j is shifted by its parent's value to the

right. A parallel split, cf. lower left part in Fig. 2, is treated as two sequences sharing the same parent node.

The *max* relation that is required for joining branches at synchronization points, cf. lower right in Fig. 2 is defined as follows. Let X_i and X_j be the parents of X_k, such that X_k is the maximum of its parents. Then, $P(X_k \leq x \mid X_i, X_j) = P(max(X_i, X_j) \leq x) = P(X_i \leq x) \cdot P(X_j \leq x) = D_i(x) \cdot D_j(x)$, i.e., the probability distribution functions are multiplied. This proves to be a challenge, as the maximum of two normally distributed random variables is no longer normally distributed. We use a linear approximation, as described in [7]. This means that we express the maximum as a normal distribution, with its parameters depending linearly on the normal distributions of the joined branches. The approximation is good, when the standard deviations of the joined distributions are similar and it degrades when they strongly diverge, cf. [7]. The resulting Bayesian network model is a linear Gaussian model, which is a class of continuous type Bayesian networks, where inference is efficiently possible, i.e., in $O(n^3)$.

Once we determined probable values for the timestamps of all missing events in a trace, we can proceed with the next trace starting another iteration of the algorithm.

3 Conclusion

Here, we presented a method to repair timed event logs in order to make them available for further analysis, e.g., with process mining tools. The formal specification, and evaluation results can be found in [5]. The method works by decomposing the problem into two sub-problems: (i) repairing the structure, and (ii) repairing the time.

This work can be considered as the first step towards eliciting a SPN model from logs with *missing data* in a *maximum likelihood* or *multiple imputation* fashion. This way, allowing to take all the observed data into account and get efficient estimations for the activity durations and path probabilities.

References

1. van der Aalst, W., et al.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part I. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012)
2. Rogge-Solti, A., van der Aalst, W., Weske, M.: Discovering Stochastic Petri Nets with Arbitrary Delay Distributions From Event Logs. In: BPM Workshops. Springer (to appear)
3. Marsan, M.A., Conte, G., Balbo, G.: A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. ACM TOCS 2(2), 93–122 (1984)
4. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann (1988)
5. Rogge-Solti, A., Mans, R., van der Aalst, W., Weske, M.: Repairing Event Logs Using Stochastic Process Models. Technical Report 78, Hasso Plattner Institute (2013)
6. Adriansyah, A., van Dongen, B., van der Aalst, W.: Conformance Checking using Cost-Based Fitness Analysis. In: EDOC 2011, pp. 55–64. IEEE (2011)
7. Zhang, L., Chen, W., Hu, Y., Chen, C.: Statistical Static Timing Analysis With Conditional Linear MAX/MIN Approximation and Extended Canonical Timing Model. In: TCAD, vol. 25, pp. 1183–1191. IEEE (2006)