Decision Support Systems 79 (2015) 138–149

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Process mining on noisy logs — Can log sanitization help to improve performance?

Hsin-Jung Cheng [a], Akhil Kumar [b,*]

[a] Department of Industrial Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan
[b] Department of Supply Chain and Information Systems, Smeal College of Business, The Pennsylvania State University, University Park, PA 16802, USA

ARTICLE INFO

Article history:
Received 4 November 2014
Received in revised form 10 August 2015
Accepted 12 August 2015
Available online 21 August 2015

Keywords:
Process mining
Benchmarking
Noisy data
Log sanitization
Metrics
Rules

ABSTRACT

Process mining techniques are designed to read process logs and extract process models from them. However, real world logs are often noisy and such logs produce bad, spaghetti-like process models. We propose a technique to sanitize noisy logs by first building a classifier on a subset of the log, and applying the classifier rules to remove noisy traces from the log. The improvement in the quality of the resulting process models is evaluated on synthetic logs from benchmark models of increasing complexity on both behavioral and structural recall and precision metrics. The results show that mined models produced from such preprocessed logs are superior on several evaluation metrics. They show better fidelity to the reference models, and are also more compact with fewer elements. A nice feature of the rule based approach is that it generalizes to any noise pattern since the nature of noise varies from one log to another. The rules can also be explained and may be further modified manually. We also give results from experiments with a real dataset.

1. Introduction

Process mining [1] is a technique that helps to extract process related knowledge (e.g., process models) from event logs and exploit it for further analysis. Event logs record the start and/or completion of various tasks in a process instance. The process models extracted from such logs using process mining algorithms are called “mined” models, and they describe the actual behavior of a business process. In the real world, process models have been extracted from logs in healthcare [2–4], local municipalities [5], semiconductor manufacturing [6], telephone repair [7], rental agencies [7], etc. In contrast to data mining, where the focus is on analyzing transaction data (about products, customers, sales, defects, etc.) to discover patterns and trends, process mining focuses on a different kind of patterns, i.e., those related to understanding relationships among activities in a specific business process. An improved understanding of process models through process mining leads to better decision making.

In process mining, our interest lies in understanding the structure and behavior of the relationships among activities. By analyzing process data in the form of actual process execution instances, we can discover patterns of how activities are performed with respect to one another. This analysis is very helpful in understanding the evolution of process models and improving them. It also helps in checking if an actual process in the real world (say, a medical treatment process) conforms to a given process model, and, if not, the extent to which it deviates. Process mining and data mining have similar objectives, i.e., to discover patterns and knowledge from large amounts of data. However, in data mining the patterns relate to relationships among data values (e.g., sales are higher in winter than in summer), while in process mining the patterns relate to specific ordering of activities with respect to one another (e.g., payment occurs after shipment). By discovering and combining such patterns, process mining techniques are able to generate complete process models from logs.

As a simple example of process mining, consider the log of a customer's browsing behavior. By analyzing such a log and building a process model for it, one can gain a better understanding of whether the user does single-tasking or multi-tasking among sites, how many sites she visits in one session, if she revisits the same site(s) within a session, does she browse only or interact as well (say, by posting comments, or make purchases), etc. A medical treatment log can be “mined” to discover a process model that reflects normal procedures. Deviances from this model (for instance an omitted diagnostic test prior to surgery) may indicate lapses in treatment.

Real world logs are often noisy because some of their (sub-) traces are duplicated, incomplete, inconsistent, or reflect some other incorrect behavior. These problems can result from data entry problems, faulty data collection instruments, data transmission or streaming problems and other technology limitations. Traces may be incomplete when

* Corresponding author.
E-mail addresses: deerq1211@gmail.com (H.-J. Cheng), akhilkumar@psu.edu (A. Kumar).

http://dx.doi.org/10.1016/j.dss.2015.08.003
0167-9236/© 2015 Elsevier B.V. All rights reserved.

certain events are missed. They could be incorrect because of recording errors. Further, inconsistencies can arise from naming conventions. Noise can also appear from transcription errors when events arrive in the wrong order. Sometimes infrequent correct behavior is also confused with noise. Such behavior usually indicates the execution of exceptional paths in the process.

Thus, it can often become difficult to distinguish between noise and low-frequency correct behavior in an event log resulting in a mined model with less fidelity to the real model. With enterprises maintaining repositories containing hundreds or thousands of event logs for different business processes, distinguishing noise in a log from correct behavior is a major problem.

The initial process mining algorithms were designed for handling noise-free event logs. But later works proposed algorithms to address noisy event logs. Agrawal, Gunopulos, and Leymann [8] were the first to apply process mining to a workflow management system. They proposed a method that automatically derives a formal model of a process from an event log. Also, they attempted to deal with noise through their proposed directed graph based algorithm. Noise was introduced by inserting erroneous activities in the log, not logging some activities that occurred, or reporting some activities in out of order time sequence. Hwang and Yang [9] proposed another directed graph based algorithm for modeling the existing processes automatically by a noise tackling mechanism to tolerate noise in the log.

Weijters and van der Aalst [10] and Weijters et al. [11] introduced Heuristics Miner, a heuristic-based approach for process mining that detects short loops and non-free-choice structures from noisy logs by considering all task pair dependencies and their frequencies. This method is based on keeping track of the frequencies of causal relationships between adjacent tasks in a log and deriving a process model based on these relationships after discarding relationships that occur infrequently as errors based on a threshold value. The Genetic Miner based on genetic principles of mutation and crossover for process discovery was proposed by Medeiros et al. [12], and they showed by experiments that it performs reasonably well with noisy logs. It can also detect non-local relationships that are not explicit in event logs based on its global search ability. An extensive survey of processing mining methods appears in [13].

Previous benchmarking studies for evaluating process mining algorithms have evaluated Heuristic Miner [11], Genetic Miner [12] and also compared Alpha algorithm [1], and Alpha++ algorithm [14]. Our current work was inspired by these and other efforts; however, our main focus here is on understanding the extent to which noise removal or "log sanitization" helps to improve the mined models.

Here, we first develop benchmark models of increasing complexity for evaluating the performance of various algorithms on both noise-free and noisy logs. These synthetic logs are created from a noise generation model. Next, we test the performance of various algorithms on these models on two metrics with varying amounts of noise present in the log. Finally, we train a classifier to mark noisy records in a log and test the performance of the algorithms on the sanitized logs. Thus, we can gain insights into the behavior of process mining algorithms on noisy vs. sanitized logs. To the best of our knowledge this is the first effort of its kind.

This paper is organized as follows. Section 2 presents the basic notions used to represent a Petri net (PN), the concept of process mining and a framework for process mining research. Section 3 presents benchmark metrics and process models, and results for noisy logs. Section 4 starts with a noise generation model, and then describes a rule-based algorithm for removing noise from a log to sanitize it. The experimental setup and results comparing the mined models from noisy and sanitized logs with the reference models are discussed in Section 5. Section **6** gives results from a real dataset and Section 7 provides an overview of related work and limitations of our work. Finally, Section 8 concludes the paper.

2. Preliminaries

In this section we discuss Petri nets, process mining and how we generate a synthetic noisy log.

2.1. Petri net (PN)

A Petri net (PN) is a common tool for graphically and mathematically modeling the states of concurrent, parallel, asynchronous, and distributed controls systems, e.g., a process model [15]. PNs are directed bipartite graphs with two types of nodes (i.e., places and transitions) [15]. Places and transitions are depicted as circles and rectangles, respectively. The directed arcs are used to connect two nodes of different types in a PN. The definition and related concepts of PNs are as follows:

Definition 1. Petri net [15] A Petri Net is a tuple $N_1 = (P, T, F)$ where

— P is a finite set of places.
— T is a finite set of transitions such that $P \cap T = \phi$.
— $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation).
— A place $p \in P$ is an input place of a transition $t \in T$ if and only if there exists a directed arc from p to t, i.e., $(p, t) \in F$.
— A place $p \in P$ is an output place of a transition $t \in T$ if and only if there exists a directed arc from t to p, i.e., $(t, p) \in F$.

As shown in Fig. 1, $P = \{p_1, p_2, \cdots, p_{13}\}$, $T = \{t_1, t_2, \cdots, t_{13}\}$, and F indicate all arcs that connect places and transitions. The marking (*or* state) M_1 of a PN is represented by black tokens distributed over one or more places (see p_1 in Fig. 1). A transition t is *enabled* if there is at least one token in each input place p, $(p, t) \in F$. If an enabled transition t *fires*, it removes one token from each of its input places p_1, $(p_1, t) \in F$ and generates one token in each of its output places p_2, $(t, p_2) \in F$.

Based on the causal relationships among its elements four basic structure types of a PN can be classified as sequence (SEQ), parallel (AND), exclusive-choice (XOR), and iteration (Loop) as illustrated in Fig. 1.

2.2. Process mining concept and framework

Fig. 2 shows the concept of process mining. Information systems generate a lot of data that is stored in event logs. Such logs can be analyzed to detect abnormal behavior such as errors or exceptions in the form of missing, unexpected or mistimed events. Process mining techniques aim to extract process models from event logs to gain a better understanding of the actual process which is often different from the prescribed process. An *event log* records traces showing the sequence of tasks performed for a particular execution of a process case instance. We assume that every time a single *task* or *activity* occurs only one *event* is recorded. Sometimes a trace may also show, in addition to a task, the name or Id of an agent who performs it and a timestamp. To illustrate how process mining techniques work, an event log is shown in Table 1. It contains six traces (representing cases) of a process for obtaining an industrial engineer's license in Taiwan. From these traces, one could observe that a candidate must apply for a license (A) and then pass both the "Operations Management" (B) and "Quality Management" (C) Exams in any order to pass stage 1 (D). Subsequently, anyone out of work study, operations research and ergonomics exams (E–G) must be passed to obtain the license (H). Process mining techniques are based on such reasoning with the event logs to build a process model in the form of the PN model shown in Fig. 3.