International Journal of Cognitive Informatics and Natural Intelligence
Volume 14 • Issue 2 • April-June 2020

Filtering Infrequent Behavior in Business Process Discovery by Using the Minimum Expectation

Ying Huang, Gannan Normal University, Ganzhou, China

Liyun Zhong, Gannan Normal University, Ganzhou, China

Yan Chen, South China Agricultural University, Guangzhou, China

ABSTRACT

The aim of process discovery is to discover process models from the process execution data stored in event logs. In the era of "Big Data," one of the key challenges is to analyze the large amounts of collected data in meaningful and scalable ways. Most process discovery algorithms assume that all the data in an event log fully comply with the process execution specification, and the process event logs are no exception. However, real event logs contain large amounts of noise and data from irrelevant infrequent behavior. The infrequent behavior or noise has a negative influence on the process discovery procedure. This article presents a technique to remove infrequent behavior from event logs by calculating the minimum expectation of the process event log. The method was evaluated in detail, and the results showed that its application in existing process discovery algorithms significantly improves the quality of the discovered process models and that it scales well to large datasets.

KEYWORDS

Business Process, Infrequent Events, Minimum Expectation, Process Mining

1. INTRODUCTION

Process mining refers to a family of techniques in the field of process management used to support the analyses of business processes based on event logs. Business process mining aims at the automatic construction of models that explain the behavior observed in event logs (Van et al., 2007). There are three classes of process mining techniques: process discovery, conformance checking, and process enhancement. Process discovery is based on an event log, and a new model, an a priori model, is constructed or discovered based on the low-level events. Conformance checking is used when there is an a priori model. The existing model is compared with the process event log, and the discrepancies between the log and the model are analyzed. Performance mining is used when there is an a priori model. The model is extended with new performance information, such as the processing times, cycle times, waiting times, and costs, so that the goal is not to check for conformance, but rather to improve the performance of the existing model with respect to certain process performance measures.

DOI: 10.4018/IJCINI.2020040101

During the process mining/process identification procedure, process discovery is the first step to construct the prior module, and it is often used to quickly obtain insights into the process under study (Van et al., 2016). Most process discovery algorithms assume that the event logs represent the behavior accurately, and that the logs are clean. Thus, these algorithms are designed to incorporate all of the behaviors in the event log into their resulting process model as much as possible (Huang et al., 2018). However, real event logs contain outliers, and these outliers may represent noise or infrequent behaviors (Măruşter et al., 2006). In general, noise refers to behavior that does not conform to the process specification and/or its correct execution. Infrequent behavior relates to events that may happen in exceptional cases of the process (Sani et al., 2017). Previous works show that the low levels of infrequent behavior have a detrimental effect on the quality of the models produced by various discovery algorithms, such as the heuristics miner (Weijters et al., 2011), the Fodina process discovery (Vanden et al., 2017), and the inductive miner (Leemans et al., 2013) algorithms, even though these algorithms claim to have noise-tolerant capabilities.

This paper deals with the issue of discovering high-quality process models in the presence of infrequent behavior in the event logs, that is, by filtering the event log prior to applying any particular process discovery algorithm.

The remainder of this paper is structured as follows. In Section 2, we discuss the related work, and in Section 3, we define the proposed technique and explain our proposed method. Details of the evaluation and the corresponding results are given in Section 4. Finally, Section 5 concludes the paper and discusses future work.

2. RELATED WORK

A number of outlier detection algorithms have been proposed in the data mining field. These algorithms build a data model (e.g., a statistical, linear, or probabilistic model) that describes the normal behavior and considers all data points that deviate from this model as outliers (Aggarwal et al., 2015).

In the context of temporal data, these algorithms have been extensively surveyed by Gupta et al. (2014) (for events with continuous values, known as time series) and by Chandola et al. (2012) (for events with discrete values, known as discrete sequences).

According to Gupta et al. (2014), we can classify these approaches into three major groups. The first group encompasses the approaches that deal with the problem of determining if an entire sequence of events is an outlier. These approaches either build a model from the entire dataset, i.e., from all the sequences (e.g., Budalakoti et al., 2009, Florez-Larrahondo et al., 2005, Sun et al., 2006 and Zhang et al., 2003) or subdivide the dataset into overlapping windows and build a model for each window (e.g., Hofmeyr et al., 1998, Lane et al., 1997, 1999). While the approaches in this group can, in principle, be used for filtering out the infrequent process behavior in event logs, the filtering would be too coarse grained and lead to the removal of entire traces in the log, which would impact the accuracy of the discovered process model.

Approaches in the second group identify single data points as outliers (e.g., Basu et al., 2007, Keogh et al., 2002 and Muthukrishnan et al., 2004) or sequences thereof (e.g., Yankov et al., 2008) on the basis of a data model of the normal behavior in the log, e.g., a statistical model. These approaches are not suitable since they work at the level of a single time series. To apply them to our problem, we would need to treat the entire log as a unique time series, which would lead to mixing events from different traces based on their absolute order of occurrence in the log. Another option is to treat every trace as a separate time series.

However, given that the process events are not repeated often within a trace, their relative frequency would be very low, which would lead to considering almost all the events of a trace as outliers.

Finally, approaches in the third group identify the anomalous patterns within sequences (e.g., Gwadera et al., 2005 and Keogh et al., 2002). These approaches assign an anomaly score to a pattern