Online Comparison of Streaming Process Discovery Algorithms

Kavya Baskar[1,2] and Marwan Hassani[1,3]

[1] Department of Mathematics and Computer Science
Eindhoven University of Technology, Eindhoven, The Netherlands
[2] k.baskar@student.tue.nl
[3] m.hassani@tue.nl

Abstract. In the active field of process mining, several techniques have been proposed in various areas like process discovery and conformance checking. The integration of data stream mining techniques in process mining has gained popularity in recent years. The ProM framework that enables process mining with streaming data has been advanced to support event streams in the recent past. In this paper we present a new extension that is built upon existing work related to obtaining process models from data streams within ProM. The extension enables researchers to visually compare the results of two different process discovery algorithms for a single incoming stream of events with different algorithms to deal with the data streams such as Lossy Counting with Budget, Sliding Window and Exponential Decay.

Keywords: Process mining · Event streams · ProM · Visualization

1 Introduction

We assume the reader to be acquainted with the field of process mining and refer to [5] for a detailed understanding of the field. The concept of process mining in the streaming domain has become a subject of interest among researchers over the recent years. To enable the application of process mining on event streams, several techniques have been proposed with regard to data storage, process discovery algorithms, conformance checking and event stream based process enhancement. One such development focuses on abstract representation approximations using algorithms which are designed for frequent item mining on data streams [8]. A prototypical implementation that corresponds to this development is provided in the process mining tool-kit ProM[4].

The previous architecture is a generalization and standardization of existing event stream-based process discovery algorithms and defines a computational mechanism applicable to a large class of process discovery algorithms. The generalization of the architecture allows for the inclusion of event stream-based process discovery algorithms within the framework, in the future.

[4] http://www.promtools.org/

This demo is based on this existing implementation of process mining with streaming data, in the ProM tool. Henceforth, the remainder of this demo will focus towards the ProM framework [6]. Previous implementations on the ProM tool mainly allow the user to obtain the resulting process model for only one of process discovery algorithms [7] [2]. In our proposal we provide researchers and practitioners with the ability to compare visually the resulting Petri nets from two specific process discovery algorithms. The idea of visually comparing the output of *two* different streaming algorithms with different settings or the same algorithm with different parameter settings at each time stamp is inspired by the stream clustering tab of MOA (Massive Online Analysis) framework [1] which is an open source framework for data stream mining with concept drift. The aim of this demo is to enhance the evaluation of streaming process discovery algorithms within the ProM framework by enabling an online analysis of the resulting models. Hence, the design decision was made to be able to compare only two algorithms simultaneously.

2 Architecture

ProM is an open source framework for a wide variety of process mining algorithms and techniques in the form of plug-ins. It is implemented in Java and is therefore platform independent.

A plug-in in ProM is an algorithm implementation that is of significance which agrees with the framework [5]. For this demo, we have created a new plugin based on five existing packages within the ProM code base, i.e. *Stream*, *EventStream*, *StreamAbstractRepresentation*, *StreamAlphaMiner* and *StreamInductiveMiner*. The new plugin is implemented in the package *KavyaBaskar* [6]that has dependencies on the aforementioned packages which have been modified to suit the new architecture.

The feature of the proposed architecture in this demo is the visualization of the resulting process models (Petri nets) from two different event stream-based process discovery algorithms. In [7] a standardized approach that extends ProM framework enabling the handling of streaming data is presented. In [8] an architecture is proposed which allows for the adoption of several process discovery techniques in the event stream context, making it very generic. The core of the implementation in this demo builds upon the architecture presented in [8]. The two specific process discovery algorithms chosen for this implementation are the *Alpha Miner* and the *Inductive Miner*. Alpha Miner was chosen since it is the very first miner to have been created for process discovery, and the Inductive Miner because it guarantees a sound workflow net and previous researches indicate it to be one of the best process discovery algorithms currently available. A new plug-in called *CompareStreamInductiveMinerStreamAlphaMinerAPNX-SEventReaderImpl* has been created in the ProM framework and the respective

[5] `http://www.processmining.org`
[6] `https://www.dropbox.com/s/xko9gbntjgclrhy/eclipse-workspace.rar?dl=0`

algorithms have been incorporated within this plug-in. Other combinations are currently under development.

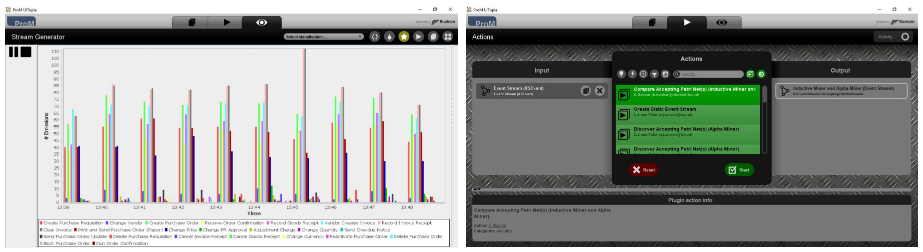
The new implementation provides the additional possibility of choosing the internal data structures for each miner- *Backward Exponential Decay*, *Forward Exponential Decay*, *Lossy Counting with Budget* and *Sliding Window* apart from the existing choice between *Frequent*, *Lossy Counting* and *Space Saving*.

The implementation of the Visualization object has been extended to split the display into two panels with process models resulting from the two different algorithms displayed on either panel. As present in earlier implementation, the Slider component at the bottom of the panel and the start/pause/stop buttons can be used to view the older model and analyze the evolution of the models. Two separate Slider components have been implemented in the new plug-in to enable the viewing of older models in any order and to be able to compare the models generated by the mining algorithms at different points of time. Moreover, the *Update Result* button can be used to generate model(s) at any random point of time as long as the stream of events are incoming.

3 Case study

As an explanatory case we have used the BPI Challenge 2017 data set[7] for the stream generation. A detailed demonstration is provided as a screen cast [8].

The log is imported and a stream is generated using the *Generate Event Stream* plug-in, setting a default emission speed of 10 data-packets/second. An example of the resulting stream is depicted in Figure 1a. A tutorial for using the



(a) Stream Generator object visualization of active stream.

(b) Applying the process discovery algorithms on the live event stream.

Fig. 1: Generating event streams and applying the algorithms

tool is available [9]. While the stream is being generated, the *CompareStreamInductiveMiner-*

[7] `https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b`

[8] `https://www.youtube.com/watch?v=2eMZe6NWhW0&feature=youtu.be`

[9] `https://www.dropbox.com/s/yh2rzymgy3higqn/Tutorial_Online_Comparison_of_Streaming_Process_Discovery_Algorithms_%28BPM_2019_Demo%29.pdf?dl=0`

StreamAlphaMinerAPNXSEventReaderImpl plug-in is selected as shown in Figure 1b, with *Event Stream (XSEvent)* as input which is created by the *Generate Event Stream* plug-in. In this plug-in, the *Case* and *Activity Identifier(s)* can be specified. From the drop down list of data structures, the *Forward Exponential Decay* has been chosen for the Inductive Miner with *Renewal Rate=1*, *Threshold=0.01* and *Decay Rate=0.01*. *Lossy Counting with Budget* for the Alpha Miner with *Budget size=1000* has been chosen. The result is depicted in Figure.2.

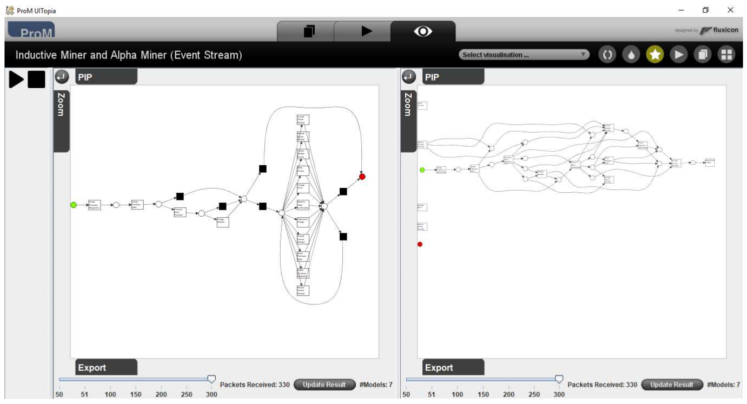


Fig. 2: Resulting Petri nets sample of two event stream-based process discovery algorithms.

4 Future Work

We plan to include conformance checking metrics such as fitness and precision comparison for both the models, visualized using a time series graph, in order to compare the performance of both algorithms. The implementation is in progress, as illustrated in Figure 3. Near future goal is to incorporate the prospect of selecting any two of the existing algorithms from the event stream-based process discovery domain(e.g. StrProM [3]), with the possibility of including future ones too (see [4] for the potential of sequential pattern mining approaches for streaming process discovery). Possibility of implementing this architecture in other streaming frameworks such as Storm, Spark, Flink, Kafka and Samza can be explored.

5 Conclusion

The newly presented simultaneous visual analysis of process models for the same event stream empowers researchers, developers and business users to experiment

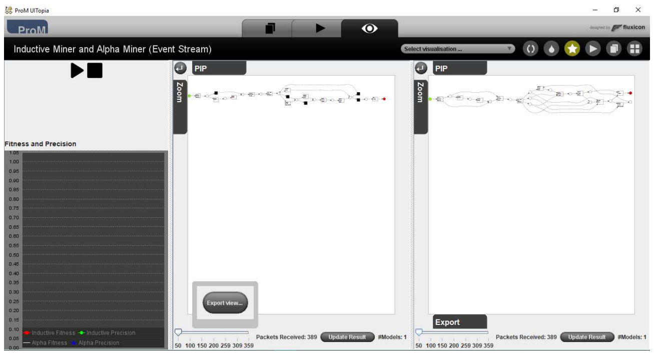


Fig. 3: Implementation of conformance checking metrics visualization.

with the concept of process discovery with streaming data within the process mining domain. The extension also allows the user to analyze the internal data structure used for handling the data stream within the ProM framework.

The lessons learned during the development of the presented implementation can be used to tackle the hurdles in building a more generic and standardized architecture which will hopefully enable the complete implementation to compare all process models resulting from algorithms of choice.

References

1. Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. MOA: Massive online analysis. *Journal of ML Research*, 11(May):1601–1604, 2010.
2. Andrea Burattin. Process mining for stream data sources. In *Process Mining Techniques in Business Environments*, pages 177–204. Springer, 2015.
3. Marwan Hassani, Sergio Siccha, Florian Richter, and Thomas Seidl. Efficient process discovery from event streams using sequential pattern mining. In *IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 1366 – 1373, 2015.
4. Marwan Hassani, Sebastiaan J. van Zelst, and Wil M. P. van der Aalst. On the application of sequential pattern mining primitives to process discovery: Overview, outlook and opportunity identification. *WIREs Data Mining and Knowledge Discovery*, e1315, 2019.
5. Wil Van Der Aalst. *Process mining: discovery, conformance and enhancement of business processes*, volume 2. Springer, 2011.
6. Wil MP van der Aalst, Boudewijn F van Dongen, Christian W Günther, RS Mans, AK Alves De Medeiros, Anne Rozinat, Vladimir Rubin, Minseok Song, HMW Verbeek, and AJMM Weijters. ProM 4.0: comprehensive support for real process analysis. In *Intl. Conf. on Application and Theory of Petri Nets*, pages 484–494, 2007.
7. Sebastiaan J van Zelst, Andrea Burattin, Boudewijn F van Dongen, and HMW (Eric) Verbeek. Data streams in ProM 6: A single-node architecture. In *BPM (Demos)*, page 81. Citeseer, 2014.
8. Sebastiaan J van Zelst, Boudewijn F van Dongen, and Wil MP van der Aalst. Event stream-based process discovery using abstract representations. *Knowledge and Information Systems*, 54(2):407–435, 2018.