# Local Process Models: Pattern Mining with Process Models

**Niek Tax, Natalia Sidorova, Wil M.P. van der Aalst** {N.TAX,N.SIDOROVA,W.M.P.V.D.AALST}@TUE.NL
Eindhoven University of Technology, The Netherlands

**Keywords**: pattern mining, process mining, business process modeling, data mining

## 1. Introduction

Process mining aims to extract novel insights from event data (van der Aalst, 2016). Process discovery plays a prominent role in process mining. The goal is to discover a process model that is representative for the set of event sequences in terms of start-to-end behavior, i.e. from the start of a case till its termination. Many process discovery algorithms have been proposed and applied to a variety of real life cases. A more conventional perspective on discovering insights from event sequences can be found in the areas of sequential pattern mining (Agrawal & Srikant, 1995) and episode mining (Mannila et al., 1997), which focus on finding frequent patterns, not aiming for descriptions of the full event sequences from start to end.

Sequential pattern mining is limited to the discovery of *sequential orderings* of events, while process discovery methods aim to discover a larger set of event relations, including sequential orderings, (exclusive) choice relations, concurrency, and loops, represented in process models such as Petri nets (Reisig, 2012), BPMN (Object Management Group, 2011), or UML activity diagrams. Process models distinguish themselves from more traditional sequence mining approaches like Hidden Markov Models (Rabiner, 1989) and Recurrent Neural Networks with their visual representation, which allows them to be used for communication between process stakeholders. However, process discovery is normally limited to the discovery of a *complete* model that captures the full behavior of process instances, and not local patterns within instances. Local Process Models (LPMs) allow the mining of patterns positioned in-between simple patterns (e.g. subsequences) and end-to-end models, focusing on a subset of the process activities and describing frequent patterns of behavior.

## 2. Motivating Example

Imagine a sales department where multiple sales officers perform four types of activities: (A) register a call for bids, (B) investigate a call for bids from the business perspective, (C) investigate a call for bids from the legal perspective, and (D) decide on participation in the call for bid. The event sequences (Figure 1(a)) contain the activities performed by one sales officer throughout the day. The sales officer works on different calls for bids and not necessarily performs all activities for a particular call himself. Applying discovery algorithms, like the Inductive Miner (Leemans et al., 2013), yields models allowing for any sequence of events (Figure 1(c)). Such "flower-like" models do not give any insight in typical behavioral patterns. When we apply any sequential pattern mining algorithm using a threshold of six occurrences, we obtain the seven length-three sequential patterns depicted in Figure 1(d) (results obtained using the SPMF (Fournier-Viger et al., 2014) implementation of the PrefixSpan algorithm (Pei et al., 2001)). However, the data contains a frequent non-sequential pattern where a sales officer first performs $A$, followed by $B$ and $C$ in arbitrary order (Figure 1(b)). This pattern cannot be found with existing process discovery or sequential pattern mining techniques. The two numbers shown in the transitions (i.e., rectangles) represent (1) the number of events of this type in the event log that fit this local process model and (2) the total number of events of this type in the event log. For example, 13 out of 19 events of type $C$ in the event log fit transition $C$, which are indicated in bold in the log in Figure 1(a). Underlined sequences indicate non-continuous instances, i.e. instances with non-fitting events in-between the events forming the instance of the local process model.

## 3. LPM Discovery Approach

A technique for the discovery of Local Process Models (LPMs) is described in detail in (Tax et al., 2016a). LPM discovery uses the process tree (Buijs et al., 2012) process model notation, an example of which is $SEQ(A, B)$, which is a sequential pattern that describes that activity $B$ occurs after activity $A$. Process tree models are iteratively ex-

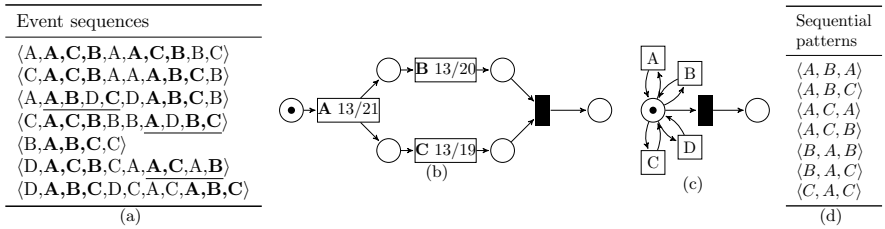


*Figure 1. (a)* A log $L$ of event sequences executed by a sales officer with highlighted instances of the frequent pattern. *(b)* The local process model showing frequent behavior in $L$. *(c)* The Petri net discovered on $L$ with the Inductive Miner algorithm (Leemans et al., 2013). *(d)* The sequential patterns discovered on $L$ with PrefixSpan (Pei et al., 2001).

panded into larger patterns using a fixed set of expansion rules, e.g., $SEQ(A, B)$ can be grown into $SEQ(A, AND(B, C))$, which indicates that $A$ is followed by both $B$ and $C$ in arbitrary order. Process trees can be converted in other process model notations, e.g., $SEQ(A, AND(B, C))$ can be converted in the Petri net of Figure 1(b). LPMs are discovered using the following steps:

1) **Generation** Generate the initial set $CM_1$ of candidate LPMs in the form of process trees.
2) **Evaluation** Evaluate LPMs in the current candidate set $CM_i$ based on support and confidence.
3) **Selection** A subset $SCM_i \subseteq CM_i$ of candidate LPMs is selected. $SM = SM \cup SCM_i$. If $SCM_i = \emptyset$ or $i \geq max\_iterations$: stop.
4) **Expansion** Expand $SCM_i$ into a set of larger, expanded, candidate process models, $CM_{i+1}$. Goto step 2 using the newly created candidate set $CM_{i+1}$.

## 4. Faster LPM Discovery by Clustering Activities

The discovery of Local Process Models (LPMs) is computationally expensive for event logs with many unique activities (i.e. event types), as the number of ways to expand each candidate LPM is equal to the number of possible process model structures with which it can be expanded times the number of activities in the log. (Tax et al., 2016b) explores techniques to cluster the set of activities, such that LPM discovery can be applied per activity cluster instead of on the complete set of events, leading to considerable speedups. All clustering techniques operate on a *directly-follows graph*, which shows how frequently the activity types of the directly follows each other in the event log. Three clustering techniques have been compared: *entropy-based clustering* clusters the activities of the directly-follows graph using an information theoretic approach. *Maximal relative information gain clustering* is a variant on entropy-based clustering. The third clustering technique uses Markov clustering (van Dongen, 2008), an out-of-the-box graph clustering technique, to cluster the activities in the directly-follows graph.

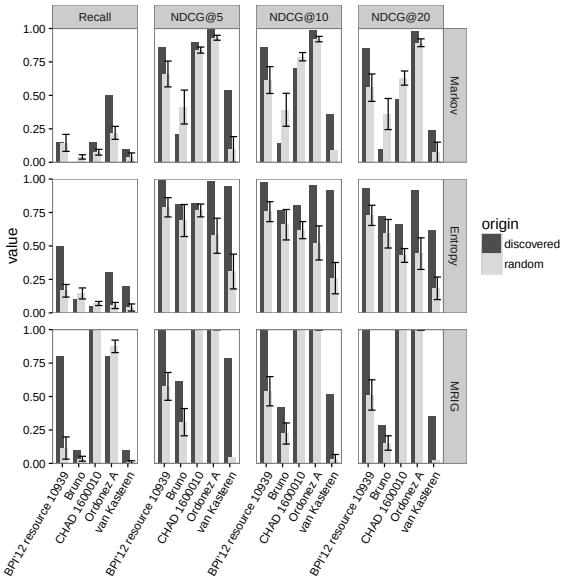


*Figure 2.* Performance of the three projection set discovery methods on the six data sets on the four metrics

We compare the quality of the obtained ranking of LPMs after clustering the activities with the ranking of LPMs obtained on the original data set. To compare the rankings we use NDCG, an evaluation measure for rankings frequently used in the information retrieval field. Figure 2 shows the results of the three clustering approaches on five data sets. All three produce better than random projections on a variety of data sets. Projection discovery based on Markov clustering leads to the highest speedup, while higher quality LPMs can be discovered using a projection discovery based on log statistics entropy. The Maximal Relative Information Gain based approach to projection discovery shows unstable performance with the highest gain in LPM quality over random projections on some event logs, while not being able to discover any projection smaller than the complete set of activities on some other event logs.

## References


Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. *Proceedings of the 11th International Conference on Data Engineering (ICDE)* (pp. 3–14). IEEE.

Buijs, J. C. A. M., van Dongen, B. F., & van der Aalst, W. M. P. (2012). A genetic algorithm for discovering process trees. *Proceedings of the 2012 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1–8).

Fournier-Viger, P., Gomariz, A., Gueniche, T., Soltani, A., Wu, C.-W., & Tseng, V. S. (2014). SPMF: a java open-source pattern mining library. *The Journal of Machine Learning Research*, *15*, 3389–3393.

Leemans, S. J. J., Fahland, D., & van der Aalst, W. M. P. (2013). Discovering block-structured process models from event logs - a constructive approach. In *Application and theory of petri nets and concurrency*, 311–329. Springer.

Mannila, H., Toivonen, H., & Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, *1*, 259–289.

Object Management Group (2011). Notation (BPMN) version 2.0. *OMG Specification*.

Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., & Hsu, M.-C. (2001). PrefixSpan: mining sequential patterns efficiently by prefix-projected pattern growth. *Proceedings of the 17th International Conference on Data Engineering (ICDE)* (pp. 215–224).

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*, 257–286.

Reisig, W. (2012). *Petri nets: an introduction*, vol. 4. Springer Science & Business Media.

Tax, N., Sidorova, N., Haakma, R., & van der Aalst, W. M. P. (2016a). Mining local process models. *Journal of Innovation in Digital Ecosystems*, *3*, 183–196.

Tax, N., Sidorova, N., van der Aalst, W. M. P., & Haakma, R. (2016b). Heuristic approaches for generating local process models through log projections. *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining* (pp. 1–8). IEEE.

van der Aalst, W. M. P. (2016). *Process mining: Data science in action*. Springer.

van Dongen, S. (2008). Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, *30*, 121–141.