Title: Desire Lines in Big Data

Name: Wil M.P. van der Aalst

Affil./Addr.: Eindhoven University of Technology

Department of Mathematics and Computer Science

PO Box 513, NL-5600 MB, Eindhoven, The Netherlands

E-mail: w.m.p.v.d.aalst@tue.nl

Desire Lines in Big Data

Synonyms

process mining, business process intelligence, distributed process mining, process discovery

Glossary

Event log: multiset of traces.

Trace: sequence of events.

Event: occurrence of some discrete incident (e.g., completion of an activity).

Process mining: collection of techniques to discover, monitor and improve real processes by extracting knowledge from event data.

Process discovery: extracting process models from an event log.

Conformance checking: monitoring deviations by comparing model and log.

Definition

Processes leave footprints in information systems just like people leave footprints in grassy spaces. *Desire lines*, i.e., the tracks formed by erosion showing where people

really walk, may be very different from the formal pathways. When people deviate from the official path there is often a good reason and room for improvement. The goal of *process mining* is to *extract desire lines from event logs*, e.g., to automatically infer a process model from raw events recorded by some information system.

Process mining techniques and tools should be able to deal with *huge heterogeneous event logs*. For example, the increasing ability to record events (cf. sensor data, internet of things, remote monitoring, and service orientation) may make it infeasible to store all events over an extended period. Therefore, *on-the-fly discovery* techniques have been developed, i.e., techniques to learn process models without storing excessive amounts of events. Moreover, techniques to *distribute* process mining techniques over a network consisting of many computing nodes are being developed. The techniques exploit modern computing infrastructures and make process mining scalable. This way it is possible to discover desire lines in Big Data.

Introduction

Desire lines refer to tracks worn across grassy spaces – where people naturally walk – regardless of formal pathways (see Figure 1). A desire line emerges through erosion caused by footsteps of humans (or animals) and the width and degree of erosion of the path indicates how frequently the path is used. Typically, the desire line follows the shortest or most convenient path between two points. Moreover, as the path emerges more people are encouraged to use it, thus stimulating further erosion. Dwight Eisenhower is often mentioned as one of the persons that noted this emerging group behavior. Before becoming the 34th president of the United States, he was the president of Columbia University. When he was asked how the university should arrange the sidewalks to best interconnect the campus buildings, he suggested letting the grass grow between buildings and delay the creation of sidewalks. After some time the de-

sire lines revealed themselves. The places where the grass was most worn by people's footsteps were turned into sidewalks.

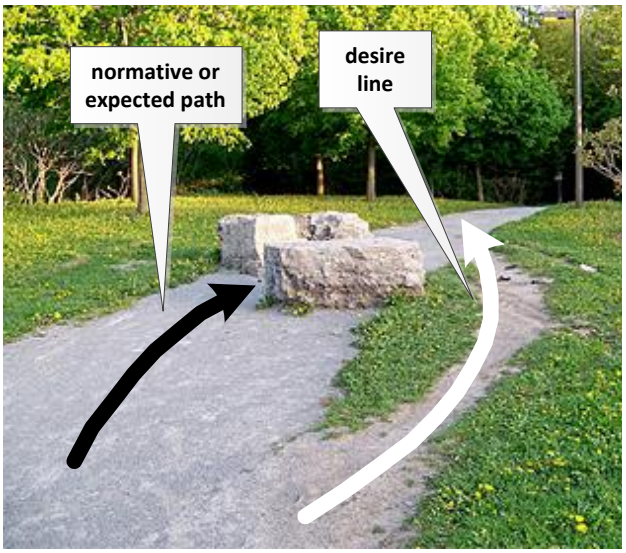


Fig. 1: Desire lines reveal the actual and not the assumed behavior of people, machines, and organizations.

The term "desire line" has been used for decades in urban planning. A desire line shows where people naturally walk. The width and degree of erosion of such an informal path indicates how frequently the path is used. Often the desire line is very different from the formal pathway. Therefore, some planners simply let erosion tell were the paths need to be. For example, the paths across Central Park in New York were reconstructed using this approach [24, 26].

Good information systems do not show signs of erosion. Nevertheless, they often contain a wealth of event data providing clues about the paths followed by the users of the system. Therefore, it is possible to determine desire lines in organizations, systems, and products. Besides visualizing such desire lines, we can also investigate how these desire lines change over time, characterize the people following a particular de-

sire line, etc. There may also be desire lines that are "undesirable" (unsafe, inefficient, unfair, etc.). Uncovering such phenomena is a prerequisite for process and product improvement.

The potential value of desire lines in "big data" (say event logs containing millions of events) is enormous. The identification of such information can be used to redesign procedures and systems ("reconstructing the formal pathways"), to recommend people taking the right path ("adding signposts were needed"), or to build in safeguards ("building fences to avoid dangerous situations").

More and more information about (business) processes is recorded by information systems in the form of so-called "event logs". IT systems are becoming more and more intertwined with these processes, resulting in an "explosion" of available data that can be used for analysis purposes. Today's information systems already log enormous amounts of events. Classical workflow management systems (e.g. FileNet, TIBCO iProcess Suite, Global 360), ERP systems (e.g. SAP, Oracle), case handling systems (e.g. BPM|one), PDM systems (e.g. Windchill), CRM systems (e.g. Microsoft Dynamics CRM, SalesForce), middleware (e.g., IBM's WebSphere, Cordys), hospital information systems (e.g., Chipsoft, Siemens Soarian), etc. provide very detailed information about the activities that have been executed. Not just information systems record data; many physical devices are connected to the Internet and objects (products and resources) are tagged and monitored. Providers of high-tech systems (ASML, Philips Healthcare, etc.) are recording terabytes of data on a daily basis. In fact, according to MGI, nearly all sectors in the US economy have at least an average of 200 terabytes of stored data per company (for companies with more than 1,000 employees) and many sectors have more than 1 petabyte in mean stored data per company [21]. Until 2000 most data was still stored in analog form (books, photos, etc.). Since 2000 data storage has grown spectacularly, shifting markedly from analog to digital [18].

Data will continue to grow at a spectacular rate. Moreover, *the digital universe and the physical universe are becoming more and more aligned*, e.g., money has become a predominantly digital entity. When booking a flight over the Internet, the customer is interacting with many organizations (airline, travel agency, bank, and various brokers), often without actually realizing it. If the booking is successful, the customer receives an e-ticket. Note that an e-ticket is basically a number, thus illustrating the tight coupling between the digital and physical universe. When the SAP system of a large manufacturer indicates that a particular product is out of stock, it is impossible to sell or ship the product even when it is available in physical form. Technologies such as RFID (Radio Frequency Identification), GPS (Global Positioning System), and sensor networks will stimulate a further alignment of data and reality, e.g., RFID tags make it possible to track and trace individual items. *Hence, there will be more and more high-quality data that can be used to reveal desire lines in any industry.*

Since we are interested in analyzing *processes* based on the data recorded, we focus on *events* that can be linked to relevant *activities*. The order of such events is important for deriving the actual process. Fortunately, most events have a timestamp or can be linked to a particular date. Hence, the event data needed for process mining are omnipresent.

Consider for example Philips Healthcare, a provider of medical systems that are often connected to the Internet to enable logging, maintenance, and remote diagnostics. For example, more than 1500 Cardio Vascular (CV) systems (i.e., X-ray machines) are monitored by Philips. On average each CV system produces 15,000 events per day, resulting in *22.5 million events per day* for just their CV systems. The events are stored for about three years and have many attributes. The error logs of ASML's lithography systems have similar characteristics and also contain about 15,000 events per machine per day. These numbers illustrate the fact that *many organizations are*

storing terabytes of event data. Earlier applications of process mining in organizations such as Philips and ASML, show that there are various *challenges* with respect to *performance* (response times), *capacity* (storage space), and *interpretation* (discovered process models may be composed of thousands of activities).

Many organizations are using so-called *Business Intelligence* (BI) software, e.g., Business Objects (SAP), Cognos (IBM), Hyperion (Oracle), etc. Common functions offered by these BI tools are reporting, online analytical processing, data mining, business performance management, benchmarks, and predictive analysis. However, these tools assume that the process is known and they typically look at data-related aspects (e.g., correlations) or view the process at an aggregate level (e.g., a dashboard showing the average response time). BI tools typically provide some form of data mining and there are dedicated data mining tools such as Weka, SPSS Clementine, RapidMiner, etc. Typical techniques supported are classification, clustering, association rules, etc. However, these systems do *not* allow for the discovery of processes based on event logs. In fact, *an explicit process notion is missing*. This led to the formation of a new research domain: *process mining*.

Key Points

The spectacular growth of event data is providing opportunities and challenges for process mining. Process discovery and conformance checking can be used to analyze and improve operational business processes in any sector. However, as event logs are growing in size it may be impossible to store, manage, and analyse event data using traditional algorithms and tools. Moreover, process mining is increasingly used on online settings where processes need to be analyzed on-the-fly. Process mining algorithms and tools need to be adapted to this new reality.

case id	event id	properties				
		timestamp	activity	resource	cost	...
1	35654423	30-12-2011:11.02	A	John	300	...
	35654424	30-12-2011:11.06	B	John	400	...
	35654425	30-12-2011:11.12	C	John	100	...
	35654426	30-12-2011:11.18	D	John	400	...
2	35655526	30-12-2011:16.10	A	Ann	300	...
	35655527	30-12-2011:16.14	C	John	450	...
	35655528	30-12-2011:16.26	B	Pete	350	...
	35655529	30-12-2011:16.36	D	Ann	300	...
...	...	...	...	...	...	...

Table 1: A fragment of some event log: each line corresponds to an event.

Process Mining

In this section, we first introduce process mining using a small example. Then we elaborate on ways to deal with huge event sets.

Process mining techniques attempt to extract non-trivial and useful information from event logs [1, 19]. One aspect of process mining is *control-flow discovery*, i.e., automatically constructing a process model (e.g., a Petri net or BPMN model) describing the causal dependencies between activities [7, 9, 29]. The basic idea of control-flow discovery is very simple: given an event log containing a set of traces, automatically construct a suitable process model "describing the behavior" seen in the log. Such discovered processes have proven to be very useful for the understanding, redesign, and continuous improvement of business processes [1].

To illustrate the notion of process discovery, consider Table 1. The table shows a small fragment of some larger event log. Only two traces are shown, both containing 4

events. Each event has a unique id and several properties. For example, event 35654423 is an instance of activity A that occurred on December 30th at 11.02, was executed by John, and costs 300 euros. The second trace starts with event 35655526 and also refers to an instance of activity A. Note that each trace corresponds to a *case*, i.e., a completed process instance.

1	$\langle A^{02}, B^{06}, C^{12}, D^{18}\rangle$
2	$\langle A^{10}, C^{14}, B^{26}, D^{36}\rangle$
3	$\langle A^{12}, E^{22}, D^{56}\rangle$
4	$\langle A^{15}, B^{19}, C^{22}, D^{28}\rangle$
5	$\langle A^{18}, B^{22}, C^{26}, D^{32}\rangle$
6	$\langle A^{19}, E^{28}, D^{59}\rangle$
7	$\langle A^{20}, C^{25}, B^{36}, D^{44}\rangle$

Table 2: A simplified event log. Each line corresponds to a trace represented as a sequence of activities with timestamps.

The information depicted in Table 1 is the typical event data that can be extracted from today's information systems. To make the example more manageable, we now focus on the activities and their timestamps only. Table 2 shows another view on the same event log. Now each line corresponds to a process instance, e.g., the first trace $\langle A^{02}, B^{06}, C^{12}, D^{18}\rangle$ refers to a process instance where activity A was executed at time 2, activity B was executed at time 6, activity C was executed at time 12, and activity D was executed at time 18. Note that the first two traces in Table 2 correspond to the fragment shown in Table 1 (using simplified timestamps).

Using existing process mining techniques it is possible to extract a process model from Table 2. For example, by applying the α algorithm [9] we obtain the process model shown in Fig. 2. This simple Petri net model [25] describes the process that starts with

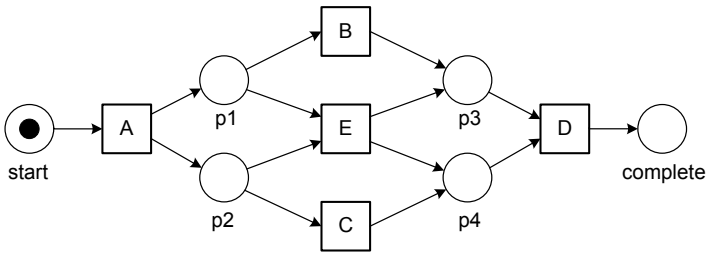


Fig. 2: A process model discovered from Table 2 using the α algorithm.

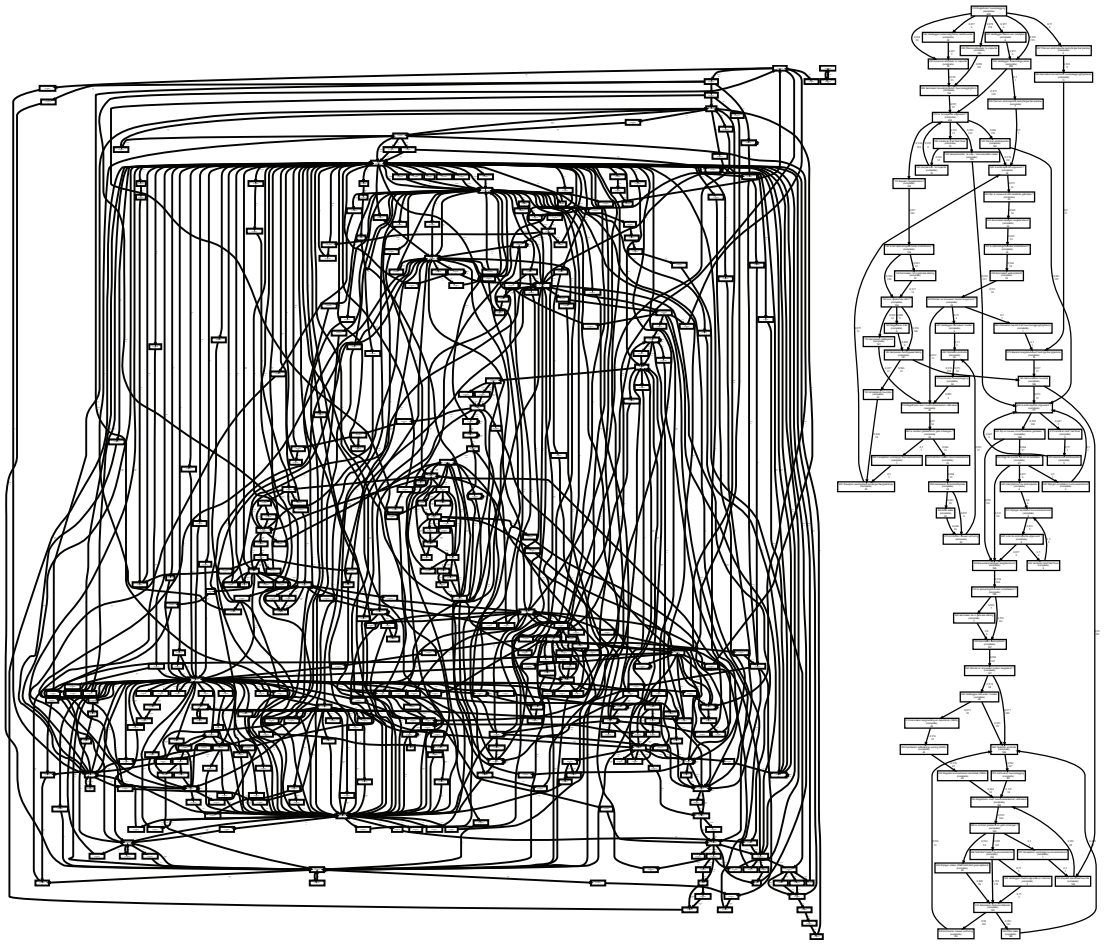
A and ends with D. In-between A and D either E or B and C are executed (in any order).

Clearly, process mining – in particular control-flow discovery – is related to the classical work on inductive inference. However, there are also notable differences because, unlike most of the classical work, process mining focuses on higher order representations which *explicitly model concurrency* (e.g., Petri nets, UML ADs, EPCs, BPMN, etc.) rather than lower level representations (e.g., Markov chains, finite state machines, or regular expressions). Moreover, we do not assume negative examples (i.e., there are no events stating that an activity *cannot* happen) and deal with issues such as incompleteness (i.e., if something did not happen, it may still be possible) and exceptional behavior. See [1] for an overview of existing process discovery approaches.

Process mining is not limited to control-flow discovery [1]. First of all, besides the *control-flow perspective* ("How?"), other perspectives such as the *organizational perspective* ("Who?") and the *case/data perspective* ("What?") may be considered. Second, process mining is not restricted to discovery. Typically three basic types of process mining are considered: (a) *discovery*, (b) *conformance*, and (c) *enhancement* [1]. In this article we will focus on process discovery, i.e., discovering a model from raw events. Discovery serves as the starting point for the two other types of process mining. The second type of process mining is *conformance* [27, 23]. Here, an existing process model is compared with an event log of the same process. Conformance checking can be used to check if reality, as recorded in the log, conforms to the model and vice versa. The

third type of process mining is *enhancement* [8]. Here, the idea is to extend or improve an existing process model using information about the actual process recorded in some event log. Whereas conformance checking measures the alignment between model and reality, this third type of process mining aims at changing or extending the a-priori model. For instance, by using timestamps in the event log one can extend the model to show bottlenecks, service levels, throughput times, and frequencies.

For example, the event log in Table 2 shows timestamps. When replaying the event log on the process model shown in Fig. 2, we can measure the time spent in the places in-between the various activities. This can be used to identify bottlenecks and predict the remaining flow time for running cases [1, 8].



(a) hospital (b) housing agency

Fig. 3: Two process models discovered using conventional process discovery techniques.

As input we assume an event log in *XES format*. In 2010, the IEEE Task Force on Process Mining standardized *XES* (`www.xes-standard.org`), a standard logging format that is extensible and supported by the *OpenXES library* (`www.openxes.org`) and by tools such as ProM, XESame, Disco, Nitro, etc. XES is the successor of the MXML format and we will also support this older format.

Fig. 3 shows two example models discovered using ProM's heuristic miner [1, 28]. The model in Fig. 3a was discovered based on event data of a group of 627 gynecological oncology patients treated in the AMC hospital in Amsterdam. All diagnostic and treatment activities have been recorded for these patients. The event log contains 24331 events referring to 376 different activities. The process model shows all 376 activities and the paths followed by patients. The model looks Spaghetti-like, but can be simplified by looking at homogeneous groups of patients and/or by focusing on the frequent activities. The model in Fig. 3b was discovered using an event log extracted from the database of a large Dutch housing agency. The event log contains 5987 events relating to 208 cases and 74 activity names. Each case corresponds to a housing unit (accommodation such as a house or an apartment). The process starts when the tenant leasing the unit wants to stop renting it. The process ends when a new tenant moves into the unit after handling all formalities.

Process Mining Challenges and Evaluation Criteria

Traditional process discovery techniques suffer from the following limitations:

- Process discovery is done *offline*, i.e., it is assumed that there is a representative event log. In some applications this assumption is unrealistic because it is impossible or too costly to store all event data. Recently, process mining techniques

have been developed for predictions and recommendations. However, also these techniques *do not discover process models on-the-fly*.

- It is impossible to discover process models for *extremely large event logs* (i.e., terabyte logs or logs with thousands of different activities). Algorithmic techniques such as heuristic mining [28], fuzzy mining [17], and the α-algorithm [9] are fast, but as data sets continue to grow even these techniques will not be able to keep up. Region-based techniques [7, 12, 29] are more precise but also time consuming. Genetic process mining algorithms [22] can be distributed easily, but are extremely inefficient.
- Most process discovery techniques *assume the process to be in steady-state*. It is assumed to be irrelevant whether a case occurs at the beginning of the log or towards the end. As a result, these techniques *do not capture concept drift* [14]. Processes may exhibit seasonal patterns (e.g., due to the increasing workload in December some checks are skipped), sudden abrupt changes (e.g., a disaster or a new law), or gradual changes (e.g., an increasing market share).
- The same process may exist within different organizations or different parts of the same organization. Within a process there may be homogeneous groups of cases that share common characteristics. Several authors proposed techniques to cluster similar cases [13, 16]. These techniques focus on producing simple models for subsets of cases. However, *the resulting process models are not related and cannot be folded easily into an overall configurable process model*.

To evaluate process models discovered using process mining, we need to *align* event log and model. Suppose that an event log contains cases that can be characterized by the following three traces: $\sigma_1 = \langle A, B, C, D \rangle$, $\sigma_2 = \langle A, C, D \rangle$, and $\sigma_3 = \langle A, C, D, B, D \rangle$. Example alignments for these three traces are (based on Fig. 2):

$$\gamma_1=\begin{array}{|c|c|c|c|}\hline A & B & C & D \\ \hline A & B & C & D \\ \hline\end{array} \quad \gamma_2=\begin{array}{|c|c|c|c|}\hline A & C & \gg & D \\ \hline A & C & B & D \\ \hline\end{array} \quad \gamma_3=\begin{array}{|c|c|c|c|c|}\hline A & C & D & B & D \\ \hline A & C & \gg & B & D \\ \hline\end{array} \quad \gamma_4=\begin{array}{|c|c|c|c|c|c|}\hline A & C & \gg & D & B & D \\ \hline A & C & B & D & \gg & \gg \\ \hline\end{array}$$

The top row of each alignment corresponds to "moves in the log" and the bottom row corresponds to "moves in the model". If a move in the log cannot be mimicked by a move in the model, then a "$\gg$" ("no move") appears in the bottom row. If a move in the model cannot be mimicked by a move in the log, then a "$\gg$" ("no move") appears in the top row. For example, in γ_1 the trace in the log (σ_1) and the model (Fig. 2) are aligned perfectly as every move in the log is mimicked by a move in the model and vice versa. In γ_2, trace σ_2 is aligned with Fig. 2. Since C is followed by D and no B occurred, the model makes a B move without a corresponding move in the log. In γ_3, trace σ_3 is aligned with Fig. 2. Now the log makes a D move without a corresponding move in the model. Given a trace in the event log, there may be many possible alignments. The goal is to find the alignment with the least number of $\gg$ elements, e.g., γ_3 seems better than γ_4. Finding a optimal alignment can be viewed as an optimization problem as shown in [5, 10].

The number of $\gg$ elements can be used to quantify *fitness*. Model and log have a perfect fitness if all traces in the log can be replayed by the model from beginning to end. Fitness is just one of the four basic conformance dimensions defined in [1]. Other quality dimensions for comparing model and log are *simplicity*, *precision*, and *generalization*.

The *simplest* model that can explain the behavior seen in the log is the best model. This principle is known as Occam's Razor. There are various metrics to quantify the complexity of a model (e.g., size, density, etc.).

The *precision* dimension is related to the desire to avoid "underfitting". It is very easy to construct an extremely simple Petri net ("flower model") that is able to

replay all traces in an event log (but also any other event log referring to the same set of activities). See [5, 23, 27] for metrics quantifying this dimension.

The *generalization* dimension is related to the desire to avoid "overfitting" [1, 5]. In general it is undesirable to have a model that only allows for the exact behavior seen in the event log. Remember that the log contains only example behavior and that many traces that are possible may not have been seen yet.

Conformance checking can be done for various reasons, e.g., to evaluate the results of process discovery. However, it may also be used to audit processes to see whether reality conforms to some normative of descriptive model [6]. Deviations may point to fraud, inefficiencies, and poorly designed or outdated procedures.

Dealing With Big Data

Figure 4 shows an overall approach for dealing with "big event data" in a comprehensive manner. Starting point are event logs that may be huge (millions of events). Events may come from different data sources that change over time. The goal is to be able to discover reliable models under these difficult circumstances. It should be possible to discover processes while storing a minimal amount of information. Moreover, for performance reasons, it should be possible to utilize a network of computers by distributing challenging process mining tasks. Processes may change over time and may vary from one organization to the other. Moreover, groups of cases may exhibit different behaviors. Therefore, it is vital to find out when and how a process changes, and how different variants of the process can be discovered and compared.

One can consider two basic approaches for on-the-fly process discovery: *sampling* and *aggregation* (see Fig. 4). For sampling we retain a representative subset of cases, e.g., based on a time window. Techniques based on aggregation do not store cases, but only aggregate information, e.g., the frequency of direct successions (with smoothing to

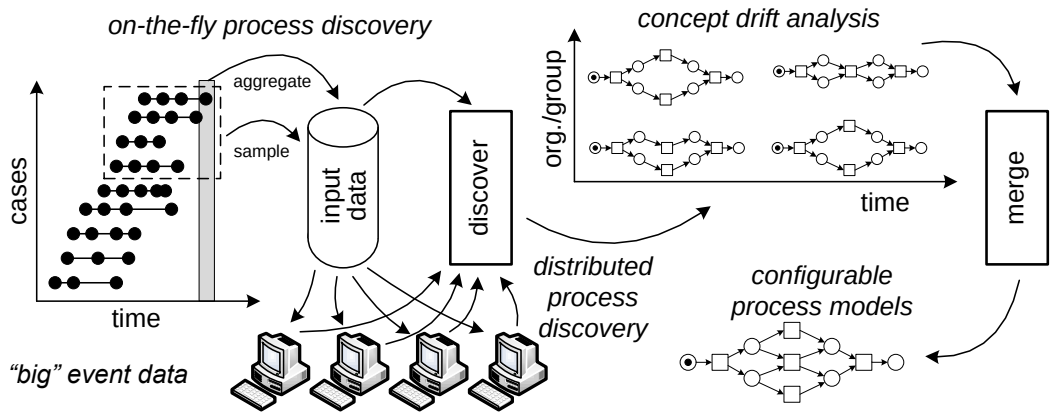


Fig. 4: Towards a more comprehensive approach to process mining supporting on-the-fly and/or distributed process mining while considering concept drift and process variability.

give more weight to recent observations). The challenge is to apply the best approach given characteristics of the log and desirable quality levels. For example, there are various tradeoffs between saving storage space and preserving model quality [15, 11].

Today, there are many different types of distributed systems, i.e., systems composed of multiple autonomous computational entities communicating through a network. Grid computing, multicore CPU systems, manycore GPU systems, cluster computing, and cloud computing all refer to systems where different resources are used concurrently to improve performance and scalability. We consider three basic types of distribution [4]. This classification is based on the way the log is partitioned.

- *Replication.* If the process mining algorithm is non-deterministic (e.g., a genetic algorithm), then the same task can be executed on all nodes and in the end the best result can be taken. In this case, the event log can be simply replicated, i.e., all nodes have a copy of the whole event log.
- *Vertical partitioning.* Event logs are composed of cases. There may be thousands or even millions of cases. These can be distributed over the nodes in the network,

i.e., each case is assigned to one computing node. All nodes work on a subset of the whole log and in the end the results need to be merged.

- *Horizontal partitioning.* Cases are composed of multiple events. Therefore, we can also partition cases, i.e., part of a case is analyzed on one node whereas another part of the same case is analyzed on another node. In principle, each node needs to consider all cases. However, the attention of one computing node is limited to a particular subset of events per case.

Process mining algorithms are typically linear in the size of the log and exponential in the number of activities. Using a vertical partitioning it is easy to achieve a linear speedup. A horizontal partitioning may be used to achieve a super linear speedup, because the time needed to solve "many smaller problems" tends to be less than the time needed to solve "one big problem" [3, 2]. This is only possible if the set of activities can be partitioned in localized process fragments. In this case, decomposition can (most likely) be used to speed up process mining algorithms even if the smaller problems are solved sequentially on just one computing node.

Processes often change while being analyzed. Therefore, *concept drift* is mentioned as one of the challenges in the Process Mining Manifesto [19]. Concept drift was been investigated in the context of various data mining problems [30, 20]. In [14] the problem is investigated in the context of process mining thereby producing some initial results. However, many challenges remain. For example, classical conformance notions such as fitness, generalization, and precision cannot be applied to processes that change [1, 5]. One needs to judge the result with respect to a moving time window of suitable length.

Key Applications

Process mining can be used to improve processes in a wide variety of organizations. A few examples of the industries were process mining has been applied.

- The *healthcare* industry includes hospitals and other care organizations. Most events are being recorded (blood tests, MRI scans, appointments, etc.) and correlation is easy because each event refers to a particular patient. The closer processes get to the medical profession, the less structured they become. For instance, most diagnosis and treatment processes tend to be rather Spaghetti-like. Medical guidelines typically have little to do with the actual processes. On the one hand, this suggests that these processes can be improved by structuring them. On the other hand, the variability of medical processes is caused by the different characteristics of patients, their problems, and unanticipated complications. Patients are saved by doctors deviating from standard procedures. However, some deviations also cost lives. Clearly, hospitals need to get a better understanding of care processes to be able to improve them. Process mining can help as event data is readily available.
- *Governments* range from small municipalities to large organizations operating at the national level, e.g., institutions managing processes related to unemployment, customs, taxes, and traffic offences. Both local and national government agencies can be seen as "administrative factories" as they execute regulations and the "products" are mainly informational or financial. Processes in larger government agencies are characterized by a high degree of automation. Consider, for example, tax departments that need to deal with millions of tax declarations. Processes in smaller government agencies (e.g., small municipalities) are typically not automated and managed by office workers rather than BPM

systems. However, due to the legal requirements, all main events are recorded in a systematic manner. Typical use cases for process mining in governments (local or non-local) are flow time reduction (e.g., shorten the time to get a building permit), improved efficiency, and compliance. Given the role of governments in society, compliance is of the utmost importance.

- *Banking* and *insurance* are two industries where BPM technology has been most effective. Processes are often automated and all events are recorded in a systematic and secure manner. Examples are the processing of loans, claims management, handling insurance applications, credit card payments, and mortgage payments. Most processes in banking and insurance are Lasagna processes, i.e., highly structured. Hence, all of the techniques presented in this book can be applied. Process discovery is less relevant for these organizations as most processes are known and documented. Typical uses cases in these industries involve conformance checking, performance analysis, and operational support.
- The *transportation* industry is also recording more and more information about the movement of people and products. Through tracking and tracing functionality the whereabouts of a particular parcel can be monitored by both sender and receiver. Although controversial, smartcards providing access to buildings and transportation systems can be used to monitor the movement of people. For example, the Dutch "ov-chipkaart" can be used to travel by train, subway, and bus. The traveler pays based on the distance between the entry point and exit point. The recorded information can be used to analyze traveling behavior. The booking of a flight via the Internet also generates lots of event data. In fact, the booking process involves only electronic activities. Note that the traveler interacts with one organization that contacts all kinds of other organizations in

the background (airlines, insurance companies, car rental agencies, etc.). All of these events are being recorded, thus enabling process mining.

These examples illustrate that there are numerous opportunities for process mining in various industries. Moreover, in all of these industries the volumes of event data will grow exponentially and there is the need to present analysis results instantly. Hence, there is a need for the distributed and on-the-fly process mining.

Future Directions

Despite the applicability of process mining there are many interesting challenges; these illustrate that process mining is a young discipline. Process discovery is probably the most important and most visible intellectual challenge related to process mining: it is far from trivial to construct a process model based on event logs that are incomplete and noisy. Still extensive research is needed to improve existing techniques or to come up with completely new techniques. Moreover, extensive research is needed to deal with "Big Data" challenges, i.e., handling event logs with millions of cases, billions of events, and thousands of different activities.

Cross References

- Data Mining
- Evolution of Social Networks
- Network Representations of Complex Data
- Role Discovery
- Service Discovery
- Temporal Networks
- Web Log Analysis

Acknowledgements

The author would like to thank all involved in the development of of the process mining tool ProM and related techniques (`processmining.org`) and all members of the IEEE Task Force on Process Mining (`www.win.tue.nl/ieeetfpm/`).

References

1. W.M.P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer-Verlag, Berlin, 2011.
2. W.M.P. van der Aalst. Decomposing Petri Nets for Process Mining: A Generic Approach. BPM Center Report BPM-12-20, BPMcenter.org, 2012.
3. W.M.P. van der Aalst. Decomposing Process Mining Problems Using Passages. In S. Haddad and L. Pomello, editors, *Applications and Theory of Petri Nets 2012*, volume 7347 of *Lecture Notes in Computer Science*, pages 72–91. Springer-Verlag, Berlin, 2012.
4. W.M.P. van der Aalst. Distributed Process Discovery and Conformance Checking. In J. de Lara and A. Zisman, editors, *International Conference on Fundamental Approaches to Software Engineering (FASE 2012)*, volume 7212 of *Lecture Notes in Computer Science*, pages 1–25. Springer-Verlag, Berlin, 2012.
5. W.M.P. van der Aalst, A. Adriansyah, and B. van Dongen. Replaying History on Process Models for Conformance Checking and Performance Analysis. *WIREs Data Mining and Knowledge Discovery*, 2(2):182–192, 2012.
6. W.M.P. van der Aalst, K.M. van Hee, J.M. van der Werf, and M. Verdonk. Auditing 2.0: Using Process Mining to Support Tomorrow's Auditor. *IEEE Computer*, 43(3):90–93, 2010.
7. W.M.P. van der Aalst, V. Rubin, H.M.W. Verbeek, B.F. van Dongen, E. Kindler, and C.W. Günther. Process Mining: A Two-Step Approach to Balance Between Underfitting and Overfitting. *Software and Systems Modeling*, 9(1):87–111, 2010.
8. W.M.P. van der Aalst, M.H. Schonenberg, and M. Song. Time Prediction Based on Process Mining. *Information Systems*, 36(2):450–475, 2011.

9. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.

10. A. Adriansyah, B. van Dongen, and W.M.P. van der Aalst. Conformance Checking using Cost-Based Fitness Analysis. In C.H. Chi and P. Johnson, editors, *IEEE International Enterprise Computing Conference (EDOC 2011)*, pages 55–64. IEEE Computer Society, 2011.

11. C. Aggarwal. *Data Streams: Models and Algorithms*, volume 31 of *Advances in Database Systems*. Springer-Verlag, Berlin, 2007.

12. R. Bergenthum, J. Desel, R. Lorenz, and S. Mauser. Process Mining Based on Regions of Languages. In G. Alonso, P. Dadam, and M. Rosemann, editors, *International Conference on Business Process Management (BPM 2007)*, volume 4714 of *Lecture Notes in Computer Science*, pages 375–383. Springer-Verlag, Berlin, 2007.

13. R.P. Jagadeesh Chandra Bose and W.M.P. van der Aalst. Trace Clustering Based on Conserved Patterns: Towards Achieving Better Process Models. In S. Rinderle-Ma, S. Sadiq, and F. Leymann, editors, *BPM 2009 Workshops, Proceedings of the Fifth Workshop on Business Process Intelligence (BPI'09)*, volume 43 of *Lecture Notes in Business Information Processing*, pages 170–181. Springer-Verlag, Berlin, 2010.

14. R.P. Jagadeesh Chandra Bose, W.M.P. van der Aalst, I. Zliobaite, and M. Pechenizkiy. Handling Concept Drift in Process Mining. In H. Mouratidis and C. Rolland, editors, *International Conference on Advanced Information Systems Engineering (Caise 2011)*, volume 6741 of *Lecture Notes in Computer Science*, pages 391–405. Springer-Verlag, Berlin, 2011.

15. A. Burattin, A.Sperduti, and W.M.P. van der Aalst. Heuristics Miners for Streaming Event Data. *CoRR*, abs/1212.6383, 2012.

16. G. Greco, A. Guzzo, L. Pontieri, and D. Saccà. Discovering Expressive Process Models by Clustering Log Traces. *IEEE Transaction on Knowledge and Data Engineering*, 18(8):1010–1027, 2006.

17. C.W. Günther and W.M.P. van der Aalst. Fuzzy Mining: Adaptive Process Simplification Based on Multi-perspective Metrics. In G. Alonso, P. Dadam, and M. Rosemann, editors, *International Conference on Business Process Management (BPM 2007)*, volume 4714 of *Lecture Notes in Computer Science*, pages 328–343. Springer-Verlag, Berlin, 2007.

18. M. Hilbert and P. Lopez. The World's Technological Capacity to Store, Communicate, and Compute Information. *Science*, 332(6025):60–65, 2011.

19. IEEE Task Force on Process Mining. Process Mining Manifesto. In F. Daniel, K. Barkaoui, and S. Dustdar, editors, *Business Process Management Workshops*, volume 99 of *Lecture Notes in Business Information Processing*, pages 169–194. Springer-Verlag, Berlin, 2012.

20. M. van Leeuwen and A. Siebes. StreamKrimp: Detecting Change in Data Streams. In *Machine Learning and Knowledge Discovery in Databases*, volume 5211 of *Lecture Notes in Computer Science*, pages 672–687. Springer-Verlag, Berlin, 2008.

21. J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. Byers. Big Data: The Next Frontier for Innovation, Competition, and Productivity. McKinsey Global Institute, 2011.

22. A.K. Alves de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst. Genetic Process Mining: An Experimental Evaluation. *Data Mining and Knowledge Discovery*, 14(2):245–304, 2007.

23. J. Munoz-Gama and J. Carmona. Enhancing Precision in Process Conformance: Stability, Confidence and Severity. In N. Chawla, I. King, and A. Sperduti, editors, *IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2011)*, pages 184–191, Paris, France, April 2011. IEEE.

24. C. Myhill. Commercial Success by Looking for Desire Lines. In *Computer Human Interaction*, volume 3101 of *Lecture Notes in Computer Science*, pages 293–304. Springer-Verlag, Berlin, 2004.

25. W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998.

26. E. Barlow Rogers. *Rebuilding Central Park: A Management and Restoration Plan*. MIT Press, 1987.

27. A. Rozinat and W.M.P. van der Aalst. Conformance Checking of Processes Based on Monitoring Real Behavior. *Information Systems*, 33(1):64–95, 2008.

28. A.J.M.M. Weijters and W.M.P. van der Aalst. Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering*, 10(2):151–162, 2003.

29. J.M.E.M. van der Werf, B.F. van Dongen, C.A.J. Hurkens, and A. Serebrenik. Process Discovery using Integer Linear Programming. *Fundamenta Informaticae*, 94:387–412, 2010.

30. G. Widmer and M. Kubat. Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning*, 23:69–101, 1996.

Recommended Reading

To get started with process mining, the reader is advised to read the book "Process Mining: Discovery, Conformance and Enhancement of Business Processes" [1] and the Process Mining Manifesto [19].