Exploring Anomalies in Time

About Temporal Deviations in Processes

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

eingereicht von
Florian Richter

München, den 28.01.2021

Erstgutachter/in:	Univ.-Prof. Dr. Thomas Seidl
Zweitgutachter/in:	Professor Andrea Burattin, Ph.D.
Drittgutachter/in:	Univ.-Prof. Dr. Stefanie Rinderle-Ma

Einreichung der Dissertation:	28.01.2021
Disputation:	22.07.2021

Eidesstattliche Versicherung

Hiermit erkläre ich, Florian Richter, an Eides statt, dass die vorliegende Dissertation ohne unerlaubte Hilfe gemäß Promotionsordnung vom 12.07.2011 angefertigt worden ist.

München, 06.08.2021	Florian Richter
Ort, Datum	Unterschrift

Abstract

Time. The nature of this very eccentric dimension has concerned humankind from the beginning. Despite culture, ethnicity, religion, or development stage, no society would have ever been emerged without making time a measurable tool to describe and control workflows. Only knowing about seasons made it possible to develop agriculture. The concept of months enables navigation and trading enterprises, and days provide a baseline for labor management. A society's technological progress is correlated with its capability to measure more adequate time intervals and observe events on a very exact timescale.

Events specify the *who*, *what*, and *when*. They establish the atomic parts of routines and tasks in our everyday world. Especially the temporal properties are critical for various event types: We arrange appointments, we wait for deliveries, we finish projects before deadlines, or we enjoy our legally assured vacation in-between periods of work. Nonetheless, the unpredictability of events provides various processes and makes the temporal perspective very exciting. On the one hand, delays often cause problems in the successive chain of events. On a personal scope, examples might be very familiar by considering, e.g., public transportation. The severity of such risks is usually higher on business scope and might inflict infringements of service level agreements like a late supply of raw materials. On the other hand, temporal deviations are advantageous in many cases. Identifying opportunities to perform tasks faster provides additional time for further actions. At this point, we should emphasize that there is no intrinsic association between acceleration/delay and benefit/detriment. E.g., a delayed train causes one person to miss an appointment and another person, that is late himself, to catch up with his schedule.

Identifying deviations in the temporal perspectives of processes provides a knowledge base for subsequent risk management operations. Since fast detection for variations is always the supreme discipline in anomaly detection, we will discuss the online discovery of process models focusing on the temporal properties of the process executions in this thesis. By identifying event-level deviations on event streams by using standardized control schemes, we discover and visualize temporal drifts in processes as Gantt charts. Then, the question arises about the impact of the execution timestamps to determine the process's footprint. Using timestamps only, we investigate the potential to perform two process mining tasks without workflow information. First, we propose an approach to match process logs, relying solely on activities' temporal occurrence and align activities of different processes utilizing the temporal behavior. Afterward, we propose a temporal conformance checking technique that applies statistical methods for the rapid generation of kernel density estimation models. Both approaches are developed as supplementary techniques to assist the workflow-based traditional methods.

We raise the anomaly detection from event to case level by introducing temporal deviation signatures as representations for the temporal characteristics of cases. The subsequent approaches use this parametrization to adapt traditional data mining techniques like clustering and outlier detection. Instead of finding singular outlier traces in processes, our focus lies on the discovery of abnormal structures. These collective anomalies are difficult to detect since standard outlier detection fails due to the missing outlier score. A first approach demonstrates that the workflow perspective already contains cluster structures regarding only the non-conforming cases of a process. We extract all non-conforming traces and apply clustering based on a geodetic distance, which refers to the process model to determine a ground distance. The discovery of substructures assists in the root-cause analysis as collective anomalies are likely to have common causes. Dealing with singular anomalies is not as efficient as solving problems for a whole class of abnormal process behavior. Therefore, we discuss the aggregation of temporally and structurally abnormal traces into anomaly micro-clusters. Adapting OPTICS for visualization and detection, we propose methods for the online monitoring of cluster structures. Thereby, process operators can be embedded in human-in-the-loop approaches while still detecting temporal deviations in a complicated and currently executed process as online monitoring.

Zusammenfassung

Zeit. Die Natur dieser sehr exzentrischen Dimension beschäftigt die Menschheit seit ihrem Anbeginn. Ungeachtet ihrer Kultur, ethnischen Zugehörigkeit, Religion oder ihres Entwicklungsstadiums basiert das Voranschreiten einer Gesellschaft auf der Messbarkeit der Zeit, denn erst dadurch werden Abläufe dokumentier- und kontrollierbar. Ohne das Wissen über Jahreszeiten wäre Landwirtschaft nicht möglich. Das Konzept einzelner Monate erlaubt die Navigation auf See und ist Voraussetzung für Handelsunternehmungen mit fernen Ländern. Tage bilden die Grundlage für das Arbeitsmanagement. Der technologische Fortschritt einer Gesellschaft korreliert mit ihrer Fähigkeit, Zeitintervalle feingranularer zu messen und Ereignisse auf einer immer genaueren Zeitskala zu beobachten.

Ereignisse, die das *Wer*, *Was* und *Wo* spezifizieren, sind die grundlegenden Elemente von Routinen und Aufgaben unseres Alltags. Besonders zeitliche Eigenschaften stellen einen kritischen Faktor für viele Ereignistypen dar: Wir vereinbaren Termine, warten auf Lieferungen, schließen Projekte fristgerecht ab oder genießen unseren gesetzlich geregelten Urlaub. Nichtsdestotrotz sind die zeitlichen Komponenten von Ereignissen meist unvorhersehbar und gerade dies macht diese Perspektive sehr spannend. Auf der einen Seite führen Verspätungen oft zu Problemem in der nachfolgenden Kette von Ereignissen. Auf einer persönlichen Ebene ist dies einfach nachvollziehbar, wenn man auf öffentlichen Nahverkehr angewiesen war. Für Unternehmen sind die Auswirkungen meist größer und betreffen meist den Verstoß gegen vertraglich geregelte Leistungen wie eine verspätete Rohstofflieferung. Andererseits sind zeitliche Abweichungen in vielen Fällen auch nützlich. Werden Möglichkeiten erkannt, Arbeitsschritte schneller auszuführen, kann die frei gewordene Zeit für weitere Aufgaben genutzt werden. An dieser Stelle müssen wir unterstreichen, dass Verzögerungen nicht notwendigerweise negativ bewertet werden müssen, genauso wie Beschleunigungen nicht immer vorteilhaft sind. Zum Beispiel bedeutet eine Zugverspätung für eine Person eine Verzögerung in seinem Tagesablauf, gleichzeitig kann ein ebenso verspäteter Fahrgast doch noch seinen Zug erreichen.

Die Identifikation zeitlicher Abweichungen in Prozessen liefert uns eine Wissensbasis für nachfolgende Operationen im Risikomanagement. Da eine schnelle Erkennung stets die Königsdisziplin der Anomalieerkennung sein wird, werden wir in dieser Thesis die Prozessmodellfindung unmittelbar zur Prozessausführung diskutieren und dabei den Fokus auf zeitliche Attribute legen. Beginnend auf dem Ereignislevel zeigen wir den Einsatz von standardisierten Kontrollschemata auf Ereignisströmen, durch die wir temporale Tendenzen herausarbeiten und als Gantt-Schaubilder visualisieren. Anschließend betrachten wir die Frage, wie viel Prozessinformation in den Zeitstempeln integriert ist und ob dieser potentielle Prozess-Fußabdruck für zwei klassische Prozessanalyseaufgaben genutzt werden kann: Zuerst stellen wir eine Methode vor, die sich ohne Berücksichtigung der Aktivitäten lediglich auf die Zeitstempeldaten stützt und die Aktivitäten zweier unabhängiger Prozesse abgleicht. Danach testen wir mit einem statistischen Ansatz die temporale Konformität mittels Dichteschätzern. Beide Methoden sind zur Unterstützung von traditionellen arbeitsflussbasierten Herangehensweisen entwickelt worden.

Durch die Einführung von zeitlicher Abweichungssignaturen als Repräsentanten für das temporale Verhalten der Prozessfälle heben wir die Anomalieerkennung vom Ereignislevel auf die Fallebene an. Die nachfolgenden Ansätze nutzen diese Parametrisierung, um traditionelle Datengewinnungstechniken wie Aggregation und Ausreißererkennung anzuwenden. Anstatt einzelne Ausreißer in Prozessdaten zu finden, konzentrieren wir uns auf die Entdeckung von anormalen Strukturen. Diese kollektiven Anomalien sind schwerer zu entdecken, da Standardmethoden darauf ausgelegt sind, einzelne Objekte im Gegensatz zu Gruppen zu finden. Eine Untersuchung zeigt, dass schon auf Basis der Arbeitsflussperspektive Strukturen aus gruppierten Objekten in den nichtkonformen Prozessinstanzen zu finden sind. Wir extrahieren ebendiese nichtkonformen Fälle und aggregieren diese mittels einer geodätische Distanz, die ein Prozessmodell als Referenz bzw. Grunddistanz verwendet. Die gefundenen Strukturen können dann zur Ursachenforschung für die Abweichungen verwendet werden, denn die Wahrscheinlichkeit ist größer, dass kollektive Anomalien einen gemeinsamen Grund für ihr abweichendes Verhalten haben. Das betrachten einzelner Anomalieobjekte ist wesentlich ineffizienter als die eventuelle Problemlösung für eine ganze Anomalieklasse. Daher diskutieren wir auch im Weiteren, wie zeitlich und strukturell anormale Prozessspuren zu Mikroansammlungen anhäufen. Unter der Verwendung von OPTICS zur Visualisierung und Erkennung stellen wir Methoden vor, die die Beobachtung und Kontrolle potentieller Häufungsstrukturen in Prozessströmen zur Ausführungszeit erleichtern. Dadurch können Prozessoperatoren in einer ständigen Mensch-Maschine-Interaktionsschleife komplexe zeitliche Veränderungen im Prozess rasch erkennen und fast unmittelbar eingreifen.

Contents

Chapter 1

Introduction to Temporal Process Deviations

> All we have to decide is what to do with the time that is given us.
>
> *J.R.R. Tolkien*

Why are temporal deviations of particular interest, especially in the process analysis domain? What are the indications of temporal deviations? Which archetypes of temporal deviations exist? In the following chapter, we will take a brief journey through the field of process mining while keeping our gaze focused on the temporal characteristics of the processes.

1.1 Motivation and Background

1.1.1 Process Observations

Time is a base concept of life [25]. Long before humans made technological advances like stone tools and fire, we perceived that the world around us is continually changing in the flow of time. Learning, regardless if it is natural or artificial learning, would not be possible without time. We were not here if we had not adapted to new situations if there were no previous derivable experiences in the past. We developed our society based on our finite individual existence's fundamental margin and the necessity to select appropriate actions.

Unfortunately, learning is limited to time itself. The major problem is obvious: Activities like collecting food during warm temperatures, hunting animals before the herd moves on, and reproducing during calmer periods is time-consuming. Although our lifespans increased, we would still be hunting for the game if we would not have developed a way to not only learn from our own experiences but preserve knowledge for our descendants.

Observing action sequences and formalizing them for documentation and knowledge transfer is the baseline definition for what we call process mining. Although we will never know the first process miner's name - or if she even had one - we continued developing more complex and sophisticated process models over time. We can observe recurring events for more considerable periods than only one life span with gestural and oral process descriptions. There is no need to look for berries in the winter if one prepares the tribe over summer sufficiently.

Cave paintings provided longer-lasting and more consistent logging of hunting introductions or other vital actions [58]. Parallelisms are challenging to describe orally, but paintings lead to extended techniques with dedicated roles that can execute different simultaneous actions. Annotations got greatly improved by unification using an actual script instead of drawings. However, either due to poor education or for simplicity, drawn working instructions kept their importance. Most megastructures would not have been possible to construct without the means to organize all the necessary building steps for various workers with different specializations and languages. Today, we often design instructions as illustrations if we want customers or employees to perceive the information. We teach children to read visual process models and implement the data into toy constructions made from plastic bricks.

We are still looking for better process models to carry more information or to transfer it more intuitively [15]. However, this is an ongoing process itself. The primary motivation is still unchanged: Develop models that help choose standard process actions based on previous executions. Knowledge transfer is always the focus, regardless of whether we want to teach young tribe members about hunting or sending new employees through an onboarding process.

Nowadays, our capabilities to measure and quantify the world have been advanced, and the amount of data we collect is enormous. We do not rely on orally passed knowledge anymore but established various digital archives. Since accuracy has also increased, we are now capable of tracing the slightest deviation. The challenge arises in identifying abnormal patterns, putting them into context, and creating guidance for process operators so that all process participants know what to do with the limited time they can provide.

1.1.2 Event-based Workflow Processes

The database community started many years ago to mine data from databases to derive **frequent patterns** that generalize the surrounding applications' characteristics and allow predictions on future database inventories. Agrawal et al. popularised the idea of **association rules**[2] in 1993, giving a name to an idea that has already been known for decades.

Revealing dependencies and correlations between items within a single database has been scaling since. Big data grows massive, so do the databases. Due to the constant data logging in all domains, most databases witness not only transactions. Execution points in time are stored as well. Having those data at hand, looking on singular database instances is not sufficient anymore. The relation between various objects provides more information than individual items. Especially the order of data objects raised much research interest. **Sequential pattern mining** extracts frequent patterns of consecutive objects. GSP[55], SPADE[69], and PrefixSpan[23] are popular algorithms in this field to identify subsequences with high support in databases.

Considering sets of database objects is very useful for market basket analysis [30] and similar applications that focus on static results at some checkout state, thus neglect the ordering. On the opposite, sequential patterns depend on a very strict ordering of objects. Many applications rely heavily on actions performed in sequence for technical reasons or ensure safety, fraud-prevention, or compliance.

Regarding the majority of applications, constraints on orderings are fundamental on a local scope. Specific series of actions have to be performed in a given order. E.g., goods have to be packed before they are stamped, and afterward, they will be delivered. Any other order does not technically make any sense. Globally, however, most processes are **semi-ordered accumulations of sequential actions**. Thus, while packing and shipping goods, it is not required to handle the accounting part before or after the shipment strictly. Due to efficiency reasons alone, parallelisms emerge in a vast amount of processes. Sometimes, simultaneous action sequences grew without intention and completely organic. This interim space between unordered transaction sets and action sequences contains the type of processes we discuss here, as illustrated in Fig. 1.1.

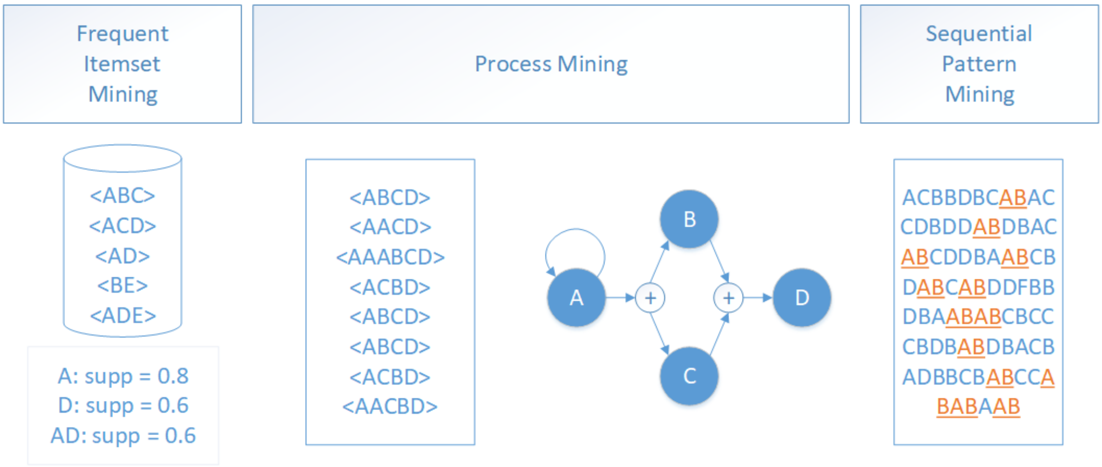
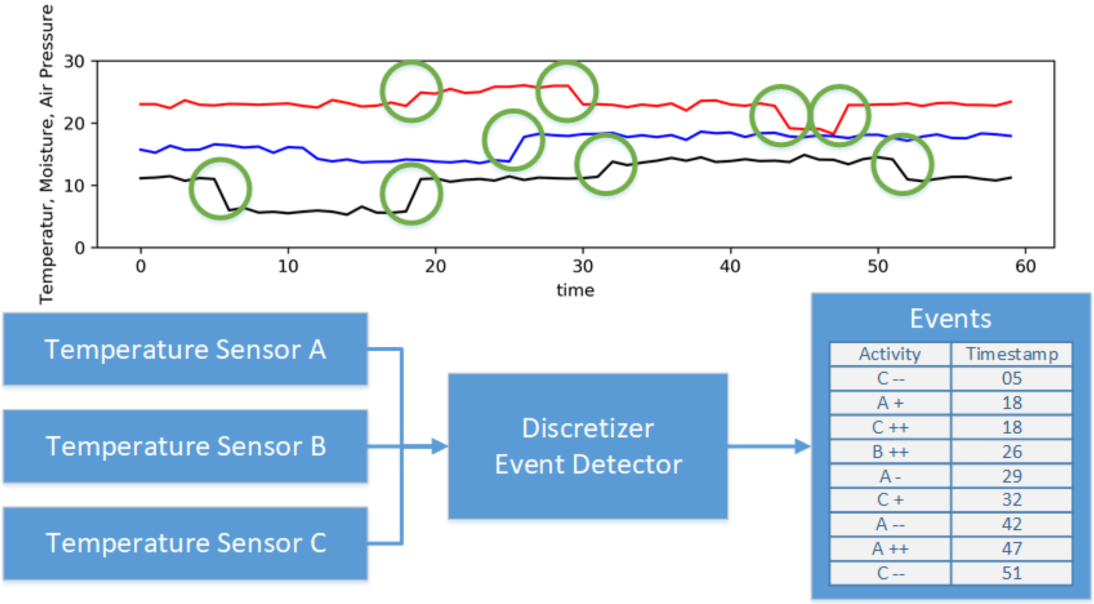


Figure 1.1: Between mining tasks neglecting sequential order and tasks that rely on strict sequences, process mining deals with partially ordered sequences and derives generalized models from sequences.

Processes contain no intrinsic definition, as the term is broadly used in almost every scientific domain. The previous description is still not sufficient to set the margin for the central topic of this thesis. The processes we are dealing with describe finite sequences of actions. These execution series represent a transformation of an object from a starting state into a final state. For instance, a typical customer journey starts with a customer's registration after requesting a product or service. Finally, he will receive the requested item, and after all debriefings, feedback cycles, and payments are made, the process execution is completed.

However, this also includes chemical or physical processes. Similarly, we start with an initial state and transform it into something different, e.g., a pile of wood is burnt to transform it into lumps of coal, smoke, and heat. While most of the discussed processes in process mining regard business-related transformations, other processes can be considered with the same techniques. We require the process to be representable by gradual events instead of a continuous movement. In particular, statistical processes often rely on measurable, continuous parameters to control the process outcome's quality. To consider such continuous world scenarios as a process mining application, we first need to redefine these processes. Every action in the process has to be translated into a discrete activity. In Fig. 1.2, we illustrate an **discretization** of a continuous sensor input into an event log. Certain increases and decreases in the temperature data indicate and change event and are represented respectively. This does only serve as an example, as the discretizer can be designed in various ways. There is a rich collection of concepts in the literature [22].



Activity	Timestamp
C --	05
A +	18
C ++	18
B ++	26
A -	29
C +	32
A --	42
A ++	47
C --	51

Figure 1.2: A continuous process can be transformed into a discrete event representation by signal processing and classification of drift prominence.

1.1.3 Events and Event Logs

Usually, we utilize proper event data and skip the discretization pre-processing steps. The baseline data sources for processes in order to perform any mining task are sets of **events**. Process operators' extended efforts to keep records of all performed operations using automated information systems lead to many data to start the mining tasks. The most dominating and driving domain for process mining is probably the business area, which uses large-scale and specialized process-aware information systems like enterprise resource planning systems [34]. Although such data sources are much more accessible for process mining, many domains provide spreadsheet collections or other modest information systems.

We do not go into detail about the actual data source formats here. Despite this, we state the requirements necessary to perform the later discussed approaches. Each event contains data about the performed action as an **activity label**. Further, we expect an event to have a **timestamp**. In most cases, the source of the event attaches this information. Otherwise, the information system registers the time at the moment of its registration. Since we focus on temporal aspects in this thesis, we demand timestamped events for the remaining chapters. We explicitly allow events with the same timestamps here since the granularity of measurement does not always provide an adequate differentiation to account for slightly shifted event occurrences.

We also require a partition of the whole event set into **cases**. A case is an execution sequence according to the process environment. Each case is assumed to perform independently. However, this constraint is always questionable. Regarding the process modeling purely as a digital simulation tool, cases are independent in theory. Since there is a physical counterpart and cases share resources, the independence assumption crumbles. Nevertheless, this assumption mirrors the desire to have processes that yield reproducible results for similar parameters. If the process provides customer service, the customer experience should not differ significantly for customers with similar profiles. Cases should use distinct case identifiers, which have usually been taken care of by the information system.

Events containing case, activity labels, and timestamps are the minimal requirements for most process mining approaches and the later-described techniques in this thesis. Regarding involved **resources** or other vital indicators, processes sometimes attach additional attributes to events. The sequence of events within a particular case is called the trace of that case. However, the terms case and trace are often used synonymously in the literature. The context usually clarifies it.

A collection of case data is called an **event log**. As previously mentioned, it might be that a case contains duplicates of events. This repetition does not have to be a database issue, but activities can reoccur within the same case. E.g., a proposal or offer is sent to a customer but not answered, so it is resent again. If activities are repeated shortly after each other, the timestamps' temporal granularity might not be large enough to guarantee specific events then. Therefore, we consider event logs as event multisets instead of event sets. Depending on the information system, ambiguities can be avoided with careful activity labeling or additional indexing.

Although an event log contains many event sequences, these sequences provide their observation's consecutive order. Single sequences do not imply that their succession is totally ordered, as discussed above. Many cases work in parallel, e.g., customers are served simultaneously by different staff members in a larger business. This concurrency continues on the case-level as well. Some activities have to be performed in order. Some can be performed in parallel. This concurrency is one of the significant properties of workflow processes. Second, particular activities or short sequences can be redone as an attempt to fix a minor defect. These loops are also not represented in individual process traces since we assume that all cases contain finitely many events. The last significant property of processes is choices in the executions. Subsections of processes can be left out without risking to fail the process execution. For instance, a customer has some options in an ordering process. The default process represents an ordering for himself. The process changes if the invoice address and the shipment address differ. We could model all different process executions individually, but we embrace the generalization of models in exchange for less accuracy in process mining. By considering multiple process cases, discovery algorithms derive process models that reveal these three properties - **concurrency, loops, and choices**[59].

1.1.4 Research on Temporal Process Features

Process mining research has a strong focus on structural aspects of processes. However, some approaches utilize the temporal aspects of events. They use it either as a standalone feature source or as an addition to the structural perspective. The most popular task depending on the related publications, is **remaining time prediction**. This task's challenge is to estimate the required time for a case to reach a final state. Depending on the process, there is a vast spectrum of possibilities to continue and complete a case. The previously collected knowledge of the already processed case prefix sometimes provides only a loose correlation with the estimated suffix.

We mention some of the works dealing with the remaining time prediction here, but we exclude related works that only focus on the next activity prediction. Van Dongen et al. used non-parametric regression to build remaining time estimators for activities and case suffixes in [63]. In [28], Leitner et al. also used regression to predict SLA violations. Many remaining time predictors or cycle-time predictors are based on process models annotated with temporal information. Starting with [62], Van der Aalst et al. used finite state machines as transition systems to replay cases and aggregate the estimated duration times on the execution paths. In [18][19], Folino et al. built predictive clustering trees first to improve the method in [62]. Rogge-Solti et al. use stochastic Petri nets to embed concurrency into the prediction to improve prediction accuracy[49]. Ceci et al. applied frequent pattern mining to extract frequent activity patterns as short-term subprocess representations[8]. These are used to predict the next activities and the completion times of the cases. Polato et al. developed a combined approach [36] using regression over case features to annotate process models. In [37], they also extend their techniques to predict the remaining times on non-stationary processes. Rogge-Solti and Weske annotated Petri nets with arbitrary distributions, allowing a more precise and individual representation of durations[50]. Navarin et al. used LSTMs to predict the remaining times[35]. Choueiri et al. included manufacturing characteristics to improve prediction results in a particular manufacturing application[12]. For a more thorough overview of publications regarding remaining time prediction, we highly recommend the survey[65] of Verenich et al.

The remaining time prediction is very prominent due to its challenge being a very well defined problem. Also, the utility is intuitive, and its business value is quite high. However, only in [43], we briefly touch this task. In the remainder, we focus on different challenges in the temporal process perspective. Different works have been proposed regarding **dura-**

tions and **temporal anomalies**. Already in 1999, Eder and Panagos[16] highlighted the importance of temporal aspects in workflow systems. Cook et al.[13] used formal timed models for temporal conformance checking by integrating time margins. Rogge-Solti and Kasneci[48] developed the first approach to apply anomaly detection in the continuous space of time instead of solely structural aspects. They used Bayesian networks to determine outlierness based on z-scored activity durations. In [5], Basile et al. proposed an approach to modify a process model to exhibit observed temporal anomalies. Böhmer and Rinderle-Ma[6] constructed likelihood graphs, similar to Bayesian networks, to model the structural component of processes in the basic variant. Extended likelihood graphs were derived to cover resource and time information. The likelihood graph is used to determine an anomaly probability for a case by traversing the graph like a decision tree. Yang et al.[68] developed a process alignment technique that considers activity durations and applies dynamic time warping to determine pairwise case distances. Lefebvre[27] proposed a technique to diagnose if observations from a stochastic timed discrete event system limited by uncomplete sensor configurations are consistent. Temporal constraints are focused on specifying tolerance intervals. Senderovich et al.[52] introduced the Temporal Network Representation based on Allen's interval algebra and illustrates pairwise temporal relations between activity executions. In [33], Mavroudopoulos et al. use distance-based methods to identify temporally abnormal process executions. Stertz et al. [56] propose temporal conformance checking as a specific task. In contrast to previous works, this raises the critical challenge on a more superior level. They rely on z-scoring of interim times between activity relations to derive a deviation score. Böhmer and Rinderle-Ma[7] developed Anomaly Detection Association Rules to detect different features in processes. Based on the Apriori algorithm for sequential pattern mining, they derive association rules. As an extension, they propose temporal ADARs. To cope with the fuzzy character of execution times, they use duration classes for generalization purposes.

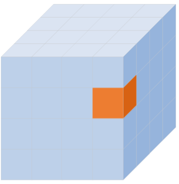
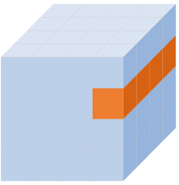
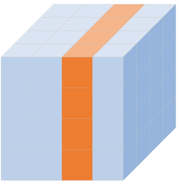
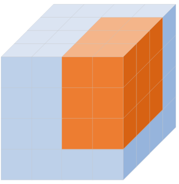
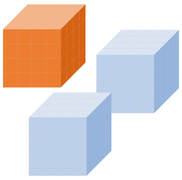
1.2 A Taxonomy of Process Anomalies

The Oxford English Dictionary defines an anomaly as "a thing, situation, etc. that is different from what is normal or expected." This definition covers a broad scope and requires sufficient clarification for every individual anomaly detection task. Hence, two major components are required: An object type to draw anomaly candidates from and a baseline representing normality or expectations.

1.2.1 Anomaly Object Types of Processes

The spectrum of object types is vast and well covered in the process mining research community. There are multiple ways to approach this classification. We pick the **data hypercube** as a starting point. In Tab. 1.1, we give a schematic overview of the relations of anomaly object types and baselines in a process application. However, these do not specify the exact representation of processes. We represent processes usually by logs or

Table 1.1: Taxonomy of process anomalies by type and baseline. For all pairs, examples are given for detection tasks or following issues. Further, related papers from this thesis are stated.

Baseline Type	**Stat. Model**	**Classifier**	**Density**	**Context**
Singleton  e.g. events	Likelihood estimation of single event existence; deviation control of event durations [42][43]	Single-point fraud detection; validity check of event features; log repair w.r.t. duplicates	Outlier detection; failure prediction in manufacturing	Data-aware outlier; event restoration w.r.t. false meta-data
Hyperpin  e.g. feature pairs	Bottleneck activity detection; resource profiling; role mining	Identifying promotion candidates; prediction of high-workload time intervals	Handling DDoS attacks; unusual frequency of activities at certain timestamp	Identifying resources with unusual demand in specific areas (season, places)
Slice  e.g. cases	Statistical conformance checking; Likelihood estimation of certain trace profiles; [44]	(Temporal) conformance checking; checks for pre-defined key performance indicators [47]	Variant mining; trace clustering; [41][40][38][39][45]	Trace clustering in distributed non-standardized systems
Subcube  e.g. subspaces	Statistical testing of process log samples	Conformance checking on episodic trace alignments	Subprocess mining; abnormal variants in specific episodes	Data-aware episode mining
Cube  e.g. processes	Process matching; log alignment; [46]	k-Nearest-Neighbor classification of process models; "Black Swan" identification	Subsidiary evaluation; comparison and performance analysis of process implementations	Season-based root-cause analysis

by process models. To match the presented anomaly taxonomy, we have to identify the according anomaly base type depending on the chosen representation.

Process data usually contains case identifiers, activity labels, and timestamps, forming a three-dimensional cube as the data space. Additional features extend the dimensionality of the data space, resulting in a hypercube. Each event in a process execution corresponds to a hypercube cell and provides the smallest or at least most individual object type for anomalies as a **singleton**. We often refer to singular event anomalies as outliers, but we have to be cautious due to overloaded terminology in the anomaly detection field. To traverse through the standard object types, we are guided by the terminologies regarding online analytical processing (OLAP)[10][60]. Event outliers are the primary target in fraud detection. As a single point of failure, either in the case of a malecious attack attempt or as a malfuntion, a single event indicates a problem before any failure cascade starts to propagate the issue.

The next considered object types for anomaly detection are the sub-dimensional slices or **hyperpins**. Fixing all but $n-2$ dimensions provides a subset of the process data. These object types are event collections projected onto smaller feature sets. E.g., we neglect case ids and consider pairs of activity and timestamp only. In this case, the object type is interesting if we look for suspicious activity occurrences at unusual times. This category contains all feature set projections that have $0 < d < n-1$ degrees of freedom. Tuples of features specify any anomaly object as hyperpins. E.g., a bottleneck analysis identifies pairs of activities and time slots with high occurrence frequency. Many role mining tasks are located here, since pairs of activities and resources are found in the result sets. Regarding frauds in this category, anomalies are typically timed attacks that abuse a service access point at the same time.

Decreasing the number of fixed dimensions and therefore increasing the degrees of freedom leads to **slices** with $n-1$ dimensions. The most prominent object type here is the case, which fixes the case identifier while carrying all remaining information as a profile. Due to its prominence, we collect anomaly detection tasks under the term conformance checking in most scenarios. Fixing specific activities or resources, like staff or machines, is applied in quality assurance or similar deviation exploration tasks.

Considering specific timestamps rarely bears sufficient information for anomaly detection since process data is often volatile. Instead, we take aggregated temporal intervals into account. In terms of OLAP, this aggregation is called roll-up. Large slices contain enough information to detect more general concepts and their abnormal counterparts. Due to the continuity of time, using slices of timestamps, we are performing concept drift detection. Aggregating resources with similar profiles is a standard job in role mining or organizational mining, and the identification of abnormal staff or machinery is also an anomaly detection task.

Regarding the hypercube structure, **subcubes** are potential anomaly candidates as well. Anomaly detection in this domain is related to frequent itemset mining or pattern mining in general if the anomalies are frequent. The rare subcube anomalies are rare patterns and represent feature associations with shared behavior and non-negative minimum occurrence. Irregular patterns lose their similarity due to the pollution by the remaining features. They are, therefore, hard to detect using a global perspective for data mining methods. Subspace clustering is also a suitable class of techniques here to identify not only process cases but, besides, the dimensions of interest for each particular cluster. E.g., the root for a temporal anomaly might inflict deviations in a certain subset of activities, but has no impact on the majority of the process. Hence, trace clustering based on the global case scope does not yield any interesting results in this case. All variants of event log samples and log projections fall into this category. However, additional refinements and compositions are possible, e.g., slicing subcubes.

These object types describe all potential objects of interest for anomaly detection in process mining regarding only one process. If we consider various processes at once, we have to deal with **multiple hypercubes**. Each process itself exists as an anomaly candidate, which we regard in the field of process matching.

1.2.2 Anomaly Baselines in Processes

Since an anomaly is a composite of an object type and a baseline, we discuss the different baselines. We make no claims of being complete here but cover the majority of types and baselines. In many cases, a strict differentiation is not possible, and margins are fuzzy. The most common baseline is a **statistical approach**. A large set of object instances is used as a training set to create a hypothesis. Statistical tests use the created distribution to determine the likeliness of tested samples. Even for populations of objects, we compare the distributions and determine the magnitude of deviation to indicate the significance of the anomaly.

Related to distributions, we can use the likelihood to derive a **classifier**. Classifiers can also be manually defined or trained in another manner. Due to the advancements in machine learning, the artificial intelligence community provides various techniques in this area. Basic classifiers start with nearest-neighbor classifiers or separating lines like support vector machines and surmount deep neural networks and complex ensemble classifiers.

The **density** baseline is used for parametrized objects. At least, a measure of similarity is necessary to establish a notion of density or sparsity. The more common approach is the identification of objects with a high dissimilarity to other objects. As said above, these are commonly known as outliers. However, anomalies sometimes occur on the other side of the density spectrum. Instead of sparsity, a cluster of similar objects is abnormal if the baseline is a sparser dataset. We call such a cluster a collective anomaly. Examples for this case are process instances that follow an atypical behavior that deviates from the remaining process. Indeed, clustering can be interpreted as an anomaly detection task since we often assume a uniform distribution as a baseline, and dense accumulations of objects are the anomalies then.

Density can be defined in various ways using distance functions if we deal with objects that contain numerical attributes. If only categorical attributes describe objects, density cannot be defined canonically. Either we define a more artificial similarity for this case or use an object's **context**. Available meta-information of an object is used to derive patterns that can be compared. A process trace might be a regular instance in one context, but changing the execution time leads to process conformance violations.

It is often possible to transform baselines and consider an anomaly from another perspective, making the **distinction somewhat fuzzy**. For instance, density in a parametrized vector space can be represented by a Gaussian mixture model. A statistical model's likelihood can be extended with a threshold, resulting in a binary classifier definition. Even for categorical attributes, we often find abstract similarity measures using binning techniques or artificial scales, e.g., customer happiness, shipping reliability, or creditworthiness. Using this representation is just a small step towards statistical models, classifiers, and density definitions.

1.3 Concept Drifts

1.3.1 Concept Drifts in Processes

Concept Drifts are of particular interest in dynamic applications like online monitoring. However, neither do all online applications contain concept drifts. Neither are concept drifts exclusive to online scenarios. Regarding the previous section, a concept drift is, first of all, an anomaly, so we have to establish a view on particular objects and a baseline first before exploring any dataset.

In machine learning, prediction models predict the properties of observed objects. If a concept drift takes place, this **prediction starts to fail** and loses accuracy. Concept drift detection is the task to localize such changing spots. Therefore, an additional requirement for concept drift detection is essential: At least one feature of the objects has to be an ordinal attribute. Otherwise, it is interpretable as such or is representable in a vector space. Since concept drift detection is mostly applied in time series analysis, concept drift is sometimes falsely reduced to this domain. However, drifts can also occur spatially or structurally. We explain by examples.

Time-based concept drifts are simple to imagine. Changes in customer behavior or regulations cause companies to adjust their business models constantly. For instance, we intuitively expect concept drifts for seasonal goods according to the changing demand over one year.

Customers and regulations do not only change globally based on the timestamp. Companies are aware that different markets have to be treated differently. Varying countries, cultures, or demographics also affect concepts, and the same model applied to different locations will fail to predict reliable results. Specific region markers like country borders or geographical units might assist in **spatial concept drift** detection, but specific problems often require appropriate cartography. Global operating enterprises with complex logistic supply chains have to take into account various concepts at once, like weather, tolls and fees, road quality, available transport types, or transport regulations. This combined risk assessment provides contour-lines that indicate concept drifts regarding risk levels and highlights regions that should be avoided. Time and space are vector spaces. The question arises if it is possible to define drifts in non-vectorial spaces. The last example considers the topological organizational structure, e.g., a company hierarchy with the directorial board on top and specialized workers with constraint tasks on the bottom. Due to the rising interest to improve environmental, social, and corporate governance ratings, identifying inequalities in ecological footprints, wages, or other concerns can be accomplished by **class concept drift detection**.

In the following, we are focusing only on the time-based scale for concept drifts. Process logs almost always provide timestamped data, while spatial information is often discarded due to privacy reasons. This argument is even more essential in the case of organizational matters. Many works consider structural changes over time in processes for concept drift detection.

1.3.2 Traditional Concept Drift Types

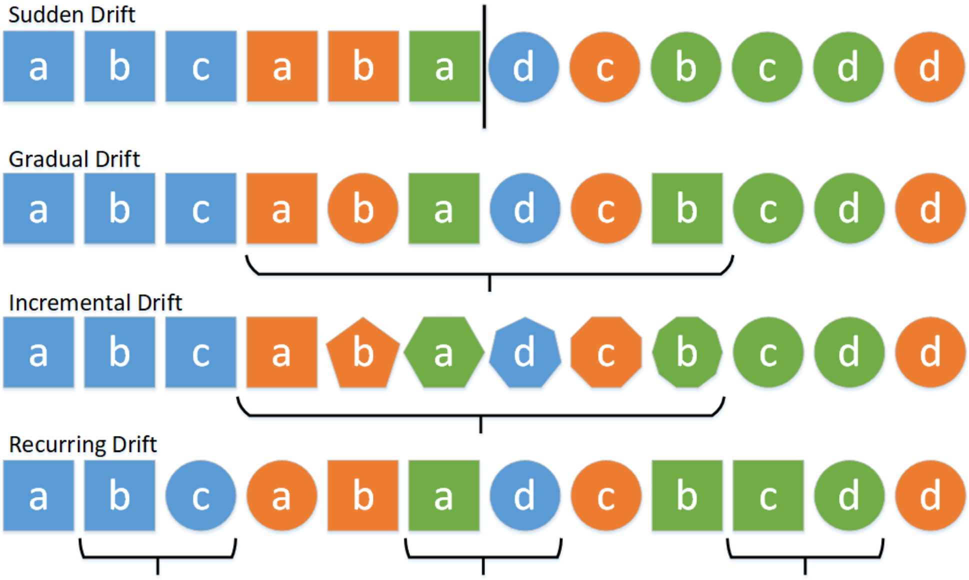


Figure 1.3: The traditional concept drift classification. Each object represents an event and colors determine cases. The objects are observed in an event stream in this order from left to right.

In the data mining community, four types of concept drifts are distinguished [20]. **Sudden drifts** occur if there is an immediate change in the predictions at the observation horizon. In general, sudden drifts are simple to detect in comparison to the other three types. Causes for sudden drifts in processes are spontaneously changing influence factors like crises, system breakdowns, or fraud attempts.

A **gradual drift** specifies a fading change over some time. Depending on the adaption speed, gradual drift detection is more challenging in general. In big data applications, the observation window is often limited. If the change is minor and the transformation period large, the observation window will likely contain only a partition of the drift. The variation within the window might be below the remaining data stream's variance, so it will not be detected. Root causes for gradual drifts are regulation changes and workflow updates. Further, concept drift detection is applied to observed data, and in all practical scenarios, we have to deal with observation latency. Hence, every sudden change does not necessarily cause a sudden drift but can affect measurable indicators with various latency times. Regarding processes, even if a change is introduced at a certain point in time, the operators adjust the different systems over time. So if a manufacturer switches the production line in the morning, some assembly machines start working on the new parts after some minutes while others have to finish previous tasks first and are changed as soon as possible.

A special case of gradual drifts is **incremental drifts**. Particularly for discrete data objects like process events, incremental drifts are very prominent in dynamic data streams. Instead of a continuous shift, a consecutive sequence of minor step-wise sudden drifts assembles an incremental concept drift. The detection is similar to sudden drift detection but similar to all concept drift detection. The magnitude of the drift determines the detection difficulty.

From a more global perspective, some concept drifts occur regularly over time. The most prominent examples are seasonal drifts like increasing ski sales in winter. In addition to the actual detection challenge of individual drifts, this task is extended to buffer previously detected drifts and compare them to recent drifts. The detection of **recurring drifts** is only possible using buffered long-term observation techniques.

1.3.3 The PSI model

In contrast to the drifts' traditional classification, we suggest another drift classification model here that indicates the **detection difficulty** of drifts. The issue with the traditional model is that it is impossible to differentiate between the drift types, especially for discrete data like events. Gradual drifts and incremental drifts behave similarly. Especially in complex processes, drifts are very likely to occur in various dimensions at once, so sudden drifts and gradual drifts cannot be distinguished. Therefore, we identify three fundamental qualitative properties for concept drifts: Pace, severity, and intention.

Since most processes are exposed to changing impacts and contain much complexity continuously, data information systems need to adapt to these changes over time. For instance, weather conditions and wear damage roads over time, slowly reducing the potential driving speed and increasing maintenance actions. This degradation causes an increase in delivery times over a long period. Although these changes are already concept drifts, detection tasks rarely aim for expected condition changes. The difficulty emerges if relevant and unexpected drifts interfere, and the drifting occurs concurrently to the ground changes. A typical example is old mining towns, which are built onto networks of tunnels. The tunneled ground causes faults, increasing the stress on the buildings above. Cracks in the walls are often misperceived as natural wear since the geological and meteorological effects have the same slow **pace**.

Severity is the most intuitive factor. Slight deviations from a determined baseline

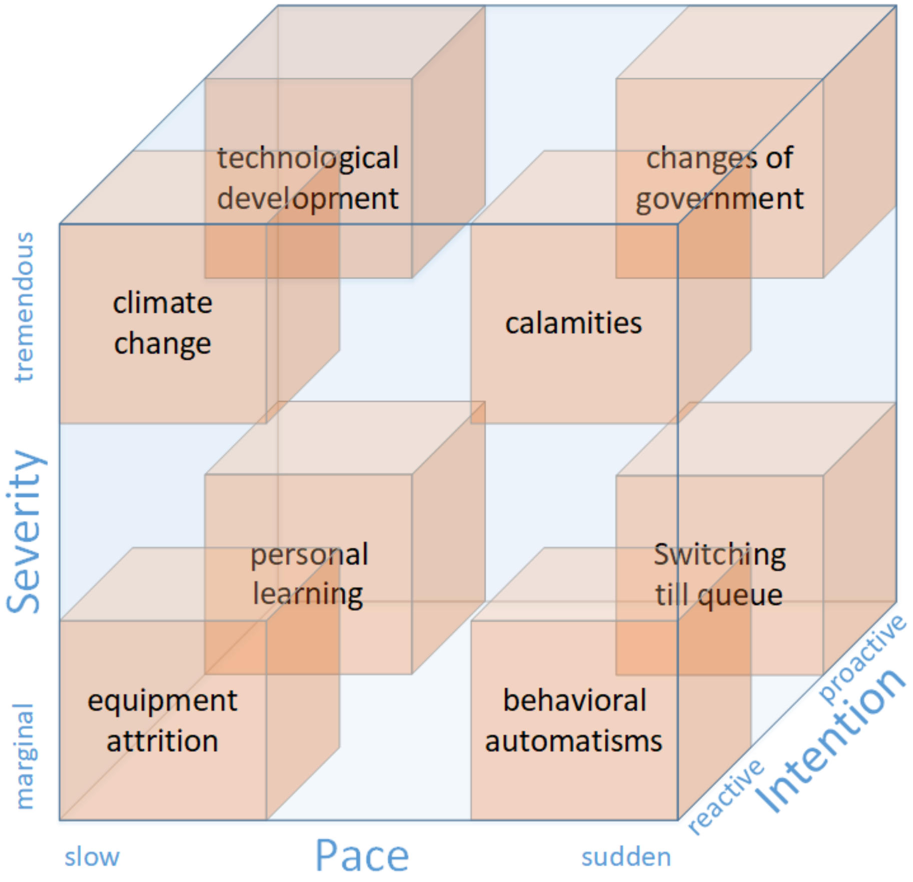


Figure 1.4: The three-dimensional PSI model with examples for the eight extreme corner marks of the model. They illustrate very extreme manifestations of pace, severity, and intention.

are more challenging to detect than profound changes. A complete process's refactoring is easily perceived in the data. A construction site in front of the receiving department is causing a few seconds delay and hence is hardly noticed.

The third drift property in this model is the **intention**. Mostly representing human intention, this property represents the expectation of drifts. Traversing this dimension, we regard process changes on the one side caused by active actions and planned to cause the effects. On the other side, we observe effects whose root causes are usually unknown and completely unexpected for the process owners. An example for the active side is an increased performance after improving machinery, while an earthquake is definitely on the opposing side.

The PSI model can be adjusted to measures of a particular process and its KPIs. In the general form as presented here, it gives the means to compare concept drifts qualitatively. In Fig. 1.4, the three-dimensional model shows some examples of concept drift classification. The higher the pace, severity, and intention of a concept drift are, the easier it is to detect it accurately. The more challenging but often more gainful detection tasks regard minor and unexpected changes over a long time.

1.3.4 Temporal Concept Drifts

However, temporal concept drift detection does not imply the observation base. The primarily addressed deviations concern the temporal perspective itself. Deviations in the temporal dimensions are either **delays or accelerations**. Delays are relatively more likely to observe than accelerations. First, most processes are already optimized to some degree beforehand, and if there are simple means to improve it further, they might have been implemented already. Second, any action requires a positive amount of time, so there is a lower limit for the duration. Although many processes implement some artificial limits to secure the process continuation, there is no theoretical upper limit. In the following chapters, we are focusing on temporal aspects only. There are quite many applications that benefit from combining structural and temporal change detection. However, our primary drive was to evaluate how well we can detect and predict different anomalies just by considering time aspects. Although there are arguments for collaborative detection techniques, both perspectives **structure and time exist orthogonally**. Anomalies do not necessarily have to occur in both views concurrently. Sometimes the results of an anomaly occur only in one perspective. Hence, we want to understand the mechanisms in the temporal perspective first. There is still the opportunity to develop ensemble techniques in future works.

Reasons for time-focused process mining techniques are various. In many cases, a structural change follows temporal deviations. E.g., important mechanical parts like the train brakes gradually lose their retarding force overtime. We never wait for the structural drift in safety-relevant systems - like a failing brake - to occur. Even if the stakes are not that high, long waiting times are often expensive. Early detections mitigate damages or expenses, but an uncovered failure can spread. E.g., faulty mechanical parts cause abrasion, which then collects in other areas and jamming consecutive parts. Eventually, the repair is not an option anymore, and a costly repurchase is required. In other scenarios, temporal drifts occur entirely without any structural response. If delays are not tied to costs or safety, temporal deviations can impact other aspects like customer satisfaction. Customers stay satisfied if they are not put into an idle state but are processed quickly.

Delays are not the only relevant deviation type. Accelerations are also exciting indicators for process changes. If a working team improved the subprocess's performance, this knowledge should be used in other areas if possible. In automated processes, acceleration is often an indicator of faulty command execution. If the execution required less time to process a certain number of objects, there might be a problem accessing them. If the failure has been unknown and unexpected, this bug has likely never been caught code-wise on the machine level. Detecting accelerations reveals fault roots, fraud issues, and potential for improvement.

1.4 Streaming Analytics for Anomaly Detection

Large enterprises produce vast amounts of data per day. Processing this data online is highly beneficial since insights can be used for improvement immediately[61]. Otherwise, data has to be collected and analyzed at a later date, which introduces some drawbacks. Online algorithms optimize four different properties[9], as seen in Fig. 1.5, and we will cover them briefly in the following.

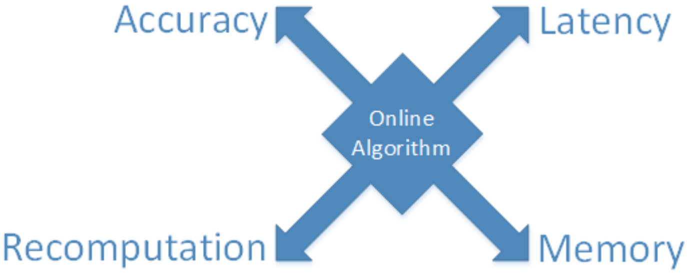


Figure 1.5: Four properties have to be balanced for every online data mining algorithm. Memory consumption and accuracy, analogously to number of computations and detection latency, are pairs of ambivalent forces, that demand an elaborated trade-off.

First of all, as also discussed previously, problems should be identified as soon as possible. The main motivation for streaming analytics is a quick response to abnormal events. Malicious untreated issues are more likely to get worse than to disappear. This effect is known as a failure cascade[14]. Hence, the first requirement for an efficient online anomaly detection approach is a **low latency** to detect anomalies in time.

To improve the detection speed, we lower the latency and rerun the algorithms more often. This naïve strategy introduces a new issue. Our resources regarding computation time are limited by hardware. Therefore, an online approach should not be applied more often than necessary. The less a recomputation is performed, the higher the latency becomes. Minimizing both latency and **number of reruns** are contradictive forces and requires a trade-off between both operations.

There is also the challenge of performing sufficient data provisioning. In an online application we only have access to the most recent objects. Depending on the used hardware, this subset can contain a large number of objects, but there will always be a finite upper limit of stored objects. Theoretically, a data stream is infinite and thus no database is capable to store all data. This **memory limitation** is the third requirement.

Analogously to the ambivalency above, memory has an opposite force. For every data mining task, we are obviously interested in accurate results. Due to the minimization of memory consumption, we do not achieve the same **accuracy** as an offline algorithm. Another trade-off between optimizing accuracy and minimizing memory consumption has to be found here as well.

These four properties are central for all online data mining algorithms. Besides, in process mining we have additional aspects to consider. First, we distinguish between different process data stream types. **Event streams** are more popular than **trace streams**. There are some works considering streams of already aggregated case data objects. In [42] and [41], we describe in detail how we collect event stream data to establish an approximate event log representation.

Another property of data streams is the **arrival rate**. The arrival of new data objects can be constant, but it is much more likely to observe a volatile arrival rate for processes. E.g., most events occur on weekdays, and fewer actions are performed on weekends. Therefore, it is more suitable to consider event chains of specific length instead of time intervals for algorithmic evaluations, e.g., regarding detection latency.

Regarding generalized data streams, the set of data items inherently contains an order. The mathematical definition of a data stream is a mapping from the natural numbers into the data object domain. Processes provide timestamped data, so the order is often determined by the event occurrence. Due to the actual processing and network message passing, both orders are not necessarily consistent[4]. We neglect this **order inconsistency** in the following, and we only utilize the timestamps attached to the arriving events.

From a theoretical perspective, better hardware does not always solve efficiency issues. The complexities of problems and algorithms dictate boundaries we cannot counter easily. **Theoretical complexity** is a mathematical classification of efficiency, which can be determined for both problems and algorithms. The problem complexity is a lower bound for any algorithm to solve a worst-case problem instance with full accuracy. The algorithmic complexity, on the other hand, states the worst-case effort to solve a problem. The complexity is given depending on the input size. In streaming analytics, the input is usually a fixed buffer of size n arrived object. Any algorithm we propose should then work at most in linear time on this buffer. Then the algorithm should finish its computation before the next buffer is overloaded. An online event stream analysis approach has to process **each event in constant time** because otherwise, the working time will build up over time, and the delay for query results will increase vastly.

Many problems have a descent complexity beyond constant or linear performance. Especially anomaly detection requires many data lookups to determine global anomalies. However, there is always a way to develop an online technique to solve a particular problem. The lower complexity is mostly traded against a less accurate result. A particular class is anytime algorithms yielding a more accurate problem solution the more time is given as a budget. A basic strategy for transforming algorithms into online techniques is **stream sampling**. A small finite chunk of data is extracted to represent the whole data stream. We achieve this compression by randomly picking stream objects or using a window containing the most recent objects.

1.4.1 Event Stream Sampling

In contrast to long-term warehousing for offline analysis, online techniques rely on a limited buffer to represent the data. The main challenge is to fill this buffer by **fairly picking data**. The sampling should not introduce any biases but should represent the complete dataset such that all properties and statistical distributions are contained. In the optimal case, the results of online algorithms applied to the sample yield the same results as offline algorithms applied to the complete dataset. This fairness is measurable if the results are comparable. However, to achieve fairness, the world is much more complicated.

There are few strict constraints for fair sampling. First, the selection strategy picks a subset of objects as representatives. Then, a **discarding strategy** removes objects from the buffer, so recent objects replace old ones. Many online algorithms choose data objects with equal probability from the data stream to ensure that all effects are covered equally and achieve fairness. For trace streams, this is a valid strategy. If we consider event streams, some issues arise. Process cases are very diverse regarding their series of activities and length. Since we have to decide whether we monitor a case or ignore it at the beginning of this case, we never know in advance if the case contains interesting anomalies or represents the majority of the data behavior.

Due to practical reasons, we usually assume cases to be finite. A process has a starting point and final goals. Cases are interrupted at some point, e.g., a repair process is usually stopped after a few cycles since, at some point, it is cheaper to produce a new item instead of repeatedly performing repair attempts. We will only consider processes with finite cases here, although there are applications with **infinite cases**. Especially in "X as a service" applications, we aim for indefinitely lasting customer journeys. Although we can subdivide such indefinite processes into finite episodic processes, long-term aspects are neglected.

A general issue regarding event stream analysis is the **lack of knowledge about when a case starts and ends**. At some point, a long-lasting case has to be discarded. Usually, we remove a case from observation after we have not received any update regarding this case. The next arriving event could continue this case, or it has been interrupted at all without any notice. Nevertheless, the buffer has a limited size, and discarding is necessary to maintain a valid sample. Therefore, after discarding a monitored case, this case's continuation is not recognized, and it will be treated as a completely new case. We already lost all previously collected case data and cannot restore it. Even events with artificial start and end tags are not sufficient to solve this problem. However, it helps to discard cases as soon as they are completed.

1.4.2 Windowing Strategies

Regarding selection and discarding strategies, we start with **landmark windows** as the most basic strategy. Until the buffer reaches its capacity, all objects are stored. When congestion is reached, all buffered data is discarded, and a new buffer is created for the following window. This method is very efficient in terms of computation time and is fair. If the window size has a suitable length, many objects are stored for the stream representation.

Applied to event streams, concurrency of cases poses significant problems. High degrees of parallelism causes much overlapping of cases. If we aggregate observed events until the buffer is full, it contains only short subtraces, and analytics are inaccurate. A better strategy is to use the finite case assumption and store a finite number of case ids. The

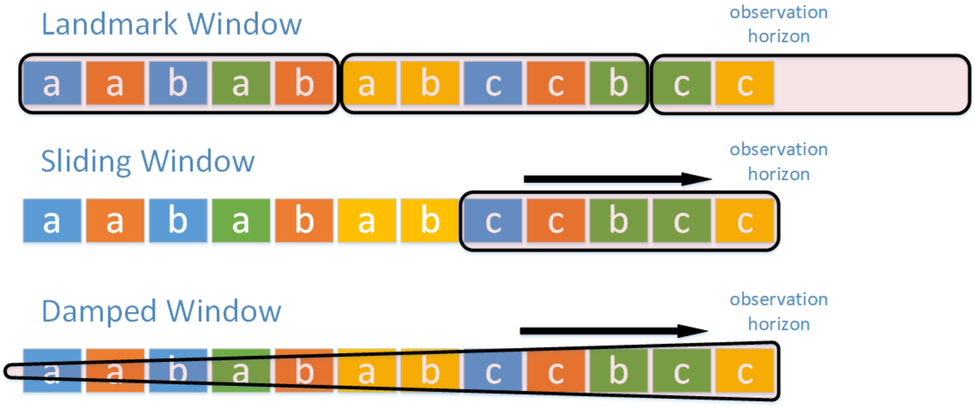


Figure 1.6: The landmark window creates new buffers at certain points in time. The sliding window moves the window iteratively and discards deprecated objects. The damped window attaches lower weights to deprecated objects, which is best used for metric data.

corresponding cases are then stored in a secondary data structure. The drawback of this strategy is a potential focus on less recent cases. The first arriving cases block the buffer. We neglect later cases if we wait for all buffered cases to become completed.

In [24], a landmark window is used. A case identifier list monitors active cases and points to intermediate data. After a predefined number of event arrivals, all data is collected and used to build a process model. The new window starts from scratch to collect new case data again. As discussed before, this works for medium concurrency processes, and the process models get inaccurate in case of more parallelism.

Even if we replace completed cases immediately with new ones, this strategy puts a strong bias on long cases since it quickly replaces short-term cases. The sample's recency improves with this **sliding window** technique, where arriving objects replace the most outdated objects continuously.

An efficient discarding strategy is crucial here since the selection of candidates has to be performed frequently. Simple queues are often utilized for sliding windows in data mining applications. Due to the unpredictable arrival order of process cases, this approach is not suitable for event streams.

TESSERACT[42][43] and OTOSO[41] apply Cuckoo hash tables. Cases are stored in a hash table, so quick replacements and look-ups are possible. Using the Cuckoo technique and assigning two potential storage slots for each case, suitable outdated cases are identified efficiently to be discarded. Cases that have not been updated for some time have a higher chance of being discarded. However, due to the replacement mechanism, there is a low chance that less recent cases will survive within the table for some time. This hash sampling provides a suitable degree of fairness while being quite efficient.

For aggregated statistics, weight is often attached to data, so it decays over time. This **damped window** approach puts the most weight on recent data, and older objects fade and disappear from the sample. This technique maintains a very recent representation while still keeping a small amount of almost outdated behavior.

Discrete objects like event data utilize tilted pyramidal discarding schemes to establish a large base of recent data objects and few outdated objects. A cascade of pushes towards the pointy end is triggered for each update, and the last spire is discarded.

AMTICS [39] establishes multi-level clusterings to derive a hierarchy of density-connected clusters. For the individual levels, micro-cluster structures store statistical data to define the cluster features. These are updated incrementally and decay overtime to forget deprecated cluster data. In the second step of TESSERACT [42][43], the anomaly baseline containing mean, and variance for various temporal relations between activities are maintained incrementally using exponentially weighted moving averages.

1.5 Collective Temporal Anomaly Detection

Singular outlier detection focuses on identifying critical anomalies that deviate from the majority of process executions. This task is critical if these individual deviations drastically disrupt the process, and their avoidance has high importance. Frauds and extreme failures are the main drivers for this discipline. Ignoring or overlooking these effects lead to high expenses. On the contrary, **collective anomalies** are very distinct from outliers and more similar to small clusters. Hence, the detection is very different. The most intuitive approach to define collective anomalies is density. The primary requirement is a distance metric between objects, which is often achieved by using vector data directly or introducing a metric representation of objects.

Regarding process cases, the objects of interest in this area do not possess a canonical vector representation. Due to the high likelihood of different case lengths, only vectorizing event data does not suffice. In [53], Song et al. proposed **trace profiles**, which transforms the trace data into vectors of equal length. The basic variant contains counts of activity occurrences as activity profiles, but they also developed an extension to embed further data like resources. These profiles are then used in various clustering techniques (k-means, quality threshold clustering, agglomerative hierarchical clustering, self-organizing maps) and using different distance functions (Euclidean, Hamming, Jaccard).

Already in [53], the authors extend their work due to the inadequate representation of the sequence data. Instead of activities, they recommend using transitions like tuples of activities to embed events' consecutive ordering. In [54], we propose profiling, which embeds temporal deviations of relations to discover patterns regarding the temporal perspective.

1.5.1 Density-Based Clustering

We define **local density** for each object as the number of neighbors below a certain distance threshold based upon a distance function. If objects exceed a minimum density threshold level, they establish a cluster. **DBSCAN**[17][51] adds all points in their proximity to this cluster and repeats this procedure cascadingly. Therefore, we do not have to specify the number of expected clusters a priori. On the other hand, the parameter choice is challenging since selecting suitable values for the **neighborhood size**, and the **minimum contained points** is not intuitive. As an extension to resolve this issue, **OPTICS**[3] compares various density levels in a hierarchical model. It highlights the data's clustering structure to determine suitable parameters for density-based clustering as a visualization tool. In Fig. 1.7, we illustrate the application of OPTICS. On the left side, points are plotted using a three blob generator model. Colored as red, green, and blue regions, we illustrate the expected clustering. OPTICS represents the dataset as a two-dimensional plot. Each object is represented on the x-axis. The height correlates with the required neighborhood size, such that this object can start a cluster. Objects are ordered by proximity to their nearest object. Hence, troughs in the plot represent higher density areas, which any density-based clustering technique yields as clusters. High dividing peaks between two troughs represent a considerable distance between both clusters. Steep ascents or descents indicate sharper boundaries of the clusters.

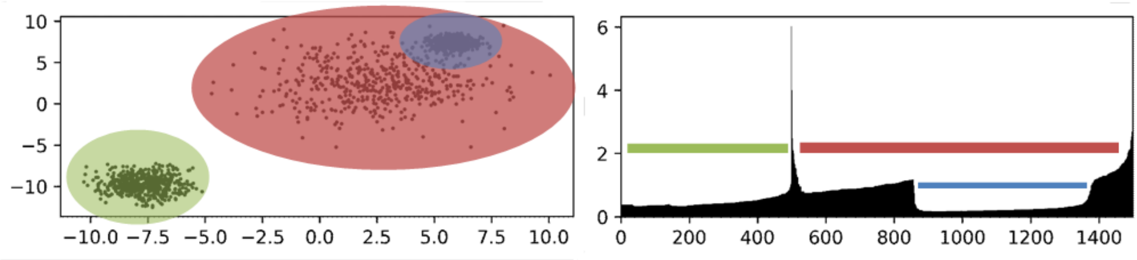


Figure 1.7: Three blob clusters are generated. Green indicates a cluster apart. The red cluster is spatially distributed and sparser than the others. Colored in blue, a dense cluster interferes with the red cluster. OPTICS visualizes this dataset on the right. We highlighted the colors above the corresponding troughs.

1.5.2 Cluster Anomalies

Regarding deviations, we assume that processes do not contain a vast number of severely abnormal cases. This assumption is typically applied to conformance checking. Cases with a high conformance score form the majority of process executions. Considering the difference between conformance scores as distance, this central partition of the process traces will cluster around the highest conformance score. Towards the lowest conformance score, the objects are sparsely distributed at an increasing rate. Process operators introduce mechanisms into processes to constrain deviations and force cases to conform to the model. However, there will always be exceptional cases, which cannot be handled according to the model. Due to their rarity, it is not practical or efficient to reduce the model‘s generalization by embedding rare deviating behavior to the model[59].

On the contrary, building individual anomaly models is a common way to deal with collective anomalies. The central property of such anomalies is a common pattern. The patterns provide a fast **classification of new objects into a particular anomaly class** instead of the binary classification between in-control and abnormal objects. In [47], we present an approach that clusters non-conforming traces regarding their structural deviation from the central process behavior. Similar deviations are identified using conformance checking, which usually highlights the deviating spot in addition to the conformance score. Establishing a distance between those deviation patterns is the challenging part. Geodetic distances determine the distance between two points based on a spatial reference system. Using the process model as a spatial map, the distances between deviating states establish this type of spatial reference model precisely. Since we define the density of traces, density-based clustering allocates the non-conforming traces into anomaly clusters. The primary motivation for collective anomaly detection and non-conforming trace clustering is the gained **efficiency** when undesired deviations occur. Instead of fixing individual and critical anomalies, a solution for a complete deviation class can be developed. After this potentially challenging step is completed, succeeding anomalies with the same characteristics are repaired without additional resources besides the actual treatment. Anomaly clustering also highlights the extends of anomalies. Especially in dynamic applications with changing environments and conditions, there are not enough resources to deal with all anomalies. However, if we cluster anomalies by identifying their characteristic patterns, we can prioritize them according to severity and likelihood. In critical situations, **risk management** relies on those insights to perform a thorough triage.

Regarding the accuracy, in collective anomaly detection, we should be careful with traditional classification accuracy. There is typically a strong focus on F1-scores in the literature. We should keep in mind that this score balances precision and recall to determine an overall accuracy[64]. Recall specifies the amount of identified anomalies versus all existing ones. Precision, on the other hand, states how many anomalies are correct in the result set. As discussed before, we apply collective anomaly detection, especially if resources are scarce. In such a scenario, identifying all potential anomalies has no priority. Finding the top-k most severe anomalies is sufficient, and the remaining candidates probably increased the detection costs while being neglected afterward. Applying detection algorithms under this premise yields the conclusion that in big data analysis, **precision is more important than recall**. This disparity is even more relevant for collective trace anomalies. Since traces with many activities provide a large degree of freedom for different deviations, anomaly clusters' borders tend to be very fuzzy. In [40], this effect is discussed more in detail. However, the center detection of such anomaly clusters is sufficient and provides the insights to perform further steps like trace anomaly classification or root-cause analysis.

1.6 Outline

The remainder of this thesis contains more in-depth discussions and technical proposals. Beforehand, we will give a **summary** to assist the reader, hopefully. The chapters contain multiple works, which are sorted by topic instead of chronological publication order. We discuss approaches to deal with different kinds of anomalies like conformance checking and clustering, emphasizing temporal aspects. In addition to a brief outline, we also give one **research question (RQ)** to highlight each approach's motivating drive.

1.6.1 Anomalous Event Detection

In the next chapter, the considered tasks comprise the online discovery of process models to derive valid activity relations. These are then observed to detect any abnormal temporal shift of their executions. An on-demand report of temporal deviations is finally provided as a Gantt chart with TESSERACT.

In StrProM[24], we collect case sequence data in a prefix tree. Using a case list to monitor currently active cases and pointing to the prefix tree nodes, we can maintain this data structure very efficiently. In a regular interval, the collected process data is transformed into intermediate data to apply the Heuristics Miner[67]. The proposed method works efficiently on an event stream and provides a landmark windowed process model of sufficient quality for further analysis tasks. Here, we investigate how well we can sample an event stream to derive process models with suitable fitness and precision (**RQ1**).

In TESSERACT[42][43], we collect event interim times, which are always possible to extract. Hence, we do not rely on specific interval event data. TESSERACT uses a modified Cuckoo Hashing technique to collect the relations online and to adjust statistics like means and variances. Each relation is observed during the event stream, and an adapting control scheme tracks deviations. By normalizing each relation using z-scoring, the collected mean and variance values modify the score to amplify stable relations and dampen noisy ones. Assuming a Gaussian distribution, a suitable approximation for the vast number of observed events, sudden drifts, and point-wise anomalies trigger an alert when they deviate by a certain number of standard deviations. We can focus on particular relations by considering a previous mined process model to show the structural relations, e.g., StrProM. Tesseract clarifies how quickly we can detect potential event duration anomalies (**RQ2**), while the journal extension evaluates how TESSERACT can be applied in remaining time prediction (**RQ3**).

1.6.2 Anomalous Trace Detection

Traversing from events to traces, we then introduce the formerly mentioned temporal deviation signatures[54]. These are then used in combination with density-based clustering to mine collective anomalies regarding the temporal perspective. Similar to TESSERACT, we collect pairwise event data and extract relation intervals. Also, we perform the same z-scoring to gain normalized noise-adjusted temporal data. This collection forms a vector space, which allows the applications of various techniques and vector distance functions. Summing up, how can we aggregate the temporal characteristics of a process case (**RQ4**)?

Regarding TOAD[40], we apply density-based clustering to temporal vector data. Following the same concept of DBSCAN[17], OPTICS[3] provides a spectrum of clusterings for any vector data. However, we do not aim for clusters since the most significant cluster in this scenario will contain all traces without any major deviation. Instead, TOAD provides a visualization to identify the structure of the temporal behavior quickly. We automatically extract small clusters of traces that are locally dense. Compared to their broader neighborhood, these traces are perceived as dense since they share a common temporal characteristic. Identifying a group of traces with shared deviations often leads to a common root-cause. Further root-cause analysis and problem-solving are efficient from an economic point of view compared to an individual analysis of each trace. TOAD answers the questions, how to detect deviating trace clusters by density (**RQ5**).

1.6.3 Quickspotting Cluster Dynamics

In the fourth chapter, we will take a look at preliminary structure detection in online applications. The tools presented here assist the previous ones by visualizing an approximation of data cohesion. A process operator receives coarse insights and can perform detailed analysis steps as a result.

AMTICS[38][39] is an extension to OPTICS. In many scenarios, the first impression of the data is all we need to decide on further steps. Do we want to start a thorough investigation, or is more explorative mining sufficient? Which are useful starting values for parameters? AMTICS returns an approximation of the cluster structure instead of an accurate result like OPTICS. On the contrary, it is applied to a data stream, and the observed density levels are adjusted anytime to react to changing conditions or focusing on specific density regions. This stream analysis approach is not designed in particular for process data and can process any metric data. In this context, however, we suggest to apply it in combination with TOAD. AMTICS elaborates on how to visualize density structure without extensive computation (**RQ6**) approximatively. The extension shows how concept drifts are tracked by AMTICS (**RQ7**).

The next logical step of cluster quickspotting is to observe such structures' dynamic lifecycles in a data stream. OTOSO[41] applies density-based clustering to temporal deviation signatures collected from event stream data. Several of the previous techniques are adapted into one technique here. The found clusters are mined and stored to derive a visualization of the structures over time. Size, density, and relations over temporal distance are visualized in one diagram, so a process owner comprehend the online process at first glance. OTOSO illustrates how structures in processes can be visualized for event streams (**RQ8**).

1.6.4 Model-Conformance-based Trace Clustering

In the fifth chapter, we perform trace clustering with unconventional distances to achieve novel insights. We use conformance scores to align trace clusters first and identifying collective anomalies of non-conform traces next.

Considering k-means, this very well-known clustering technique is admired in both research communities and industrial users. In process mining, k-means is primarily used to cluster vectorized case representations containing various cases' characteristics. k-process [45] performs, similarily to k-means, an iterative two-step approach but uses techniques from the process mining domain. Instead of using arithmetic means as cluster representatives, process models are mined. The alignment step applies conformance checks to determine the best fitting models. k-process evaluates whether techniques in the process mining domain are suitable for clustering by only relying on process-aware representations (**RQ9**).

In [47], we also perform clustering. Here, we explicitly focus on non-conforming cases only. In applications, process owners often discard non-conforming cases as undesired executions. Instead of neglecting their potential, we aim for small clusters of cases that have similar deviating behavior. The similarity is measured by using a shortest-path-based distance function on the process model. The mined collective anomalies of non-conforming cases provide a starting point for further tasks, which is more efficient than evaluating each non-conforming case's issues. The primary motivation here is how a process model can be utilized as a geodetic distance reference model for collective anomaly detection (**RQ10**).

1.6.5 Time-based Conformance Matching

In the last chapter, we return to the temporal perspective again by proposing an immediate temporal conformance checking, suitable to speed up techniques that rely on loops of repeated checks. TADE [44] is premised on kernel density estimation and uses this statistical model to estimate activity occurrence timestamps. Since KDE is a swift technique, both regarding learning models and prediction, temporal conformance checking becomes computationally very vast. TADE is the right choice for precomputation of non-conforming cases, so the remaining cases have a larger budget to be evaluated more thoroughly. We investigated how fast temporal conformance checking can be performed (**RQ11**).

Finally, we move to the process level and show how to match different processes with potential different activity labeling. Clustering and anomaly detection tasks can be performed at a very high level. Imagine having a portfolio of different processes, e.g., being a company owning different sub-companies in different sectors and varying business models. Acquiring a new asset requires the adjustment of the former process to fit into the portfolio. A matching of activity labels has to be performed. Besides, the new process's labeling originates from different sources, and matching might be impossible by relying on the activity labels. LiProMA [46] demonstrates how to use the temporal occurrences of activities for process profiling. These temporal profiles are matched using the Earth Mover's Distance, and potential label correlations are yielded. LiProMA evaluates how well do timestamps characterize processes (**RQ12**).

1.7 Conclusion

In this thesis, we illustrate the discovery of patterns and structures from the temporal process perspective. Singleton outliers and collective anomalies are derived in offline and online settings. Focusing on temporal deviations, we observed some recurring aspects we want to highlight here as **three recommendations**.

Durations have to be treated context-aware in any case. Activities behave very specifically and have different constraints and backgrounds. Hence, the activity durations and activity relation interim times might differ by several orders of magnitude. E.g., automated registration tasks require few milliseconds, while shipment tasks can take weeks to complete. We often expect that high average durations imply high variances. There is a strong perception bias since our intuition fools us by neglecting tiny temporal deviations while amplifying deviations of large absolute intervals. This differential threshold from a psychological perspective is explained by Weber's law[66]. Faulty activities with short durations can get extended by multiple times and potentially stay unnoticed, which is even a more severe issue. Long-term activities are observed more carefully. Service level agreements usually regulate the complete case duration or at least specific milestones. Therefore, these long-lasting activities have a greater impact on the completion time than short activities. To counter this bias, one shall **apply some type of standardization technique**, e.g., 0-1-normalization or z-scoring. In most cases, the number of events is sufficiently high to assume an almost Gaussian distributed domain for the timestamps. Due to durations being strictly positive, one has to be cautious with certain activities. If the average duration is smaller than the variance, the assumption fails to be a suitable approximation. However, the activity is also not a reliable deviation indicator then.

The majority of logged events are timestamped automatically. However, many processes contain manual tasks, and at some point in time, somebody has to push a button. The human interferences decrease the confidence of predictions by increasing the data variance. Different operators with daily changing conditions and minor skills in performing tasks completely reproducible are not expected to result in entirely accurate events that have durations with low variance. If the variance is the problem for a particular task, and we can neglect any creativity to perform it, we design robots to perform the job. The challenge is not to identify the human-in-the-loop as the difficulty factor for any data mining task since we are usually aware of it beforehand. The actual issue arises if we consider automated timestamps and regard them as absolutely reliable. Individual design choices introduce unwanted problems for later mining tasks and resulting in low data quality. Especially for globally operating process owners, the used information systems should be based on a mutual time reference model. Logging local time zones and daylight saving times introduces an unnecessary complexity for the duration analysis. Further, the timestamps require a suitable accuracy. Suppose the event log is only capable of storing events with a temporal precision of minutes. In that case, activities below that threshold are impossible to evaluate, and shorter loops of recurring activities cause unwanted duplicate entries. On the contrary, being too precise without necessity is also not recommended. Finally, there is often a delay between performing a task and logging it. There are technical reasons for a mismatch, e.g., network latencies. Besides technical aspects, design mismatches are often overlooked. Mismatches occur between different subprocesses or between processes of different but interacting process owners. In that case, the exact time points need additional clarification. For instance, an airline customer's transportation service has been completed when he claims his baggage at the destination airport. For the airport, the service is technically finished after the customer leaves the building. If factions regard disparate points in time for the same tasks, this matching must be resolved before any analysis can be performed. To sum this up, we advise to **never blindly trust automated timestamps**.

As a third observation, we advise to **regard any temporal analysis and exciting insights orthogonal to structural process mining results**. Effects might occur in both or more perspectives simultaneously. Of course, process owners prefer an aggregated picture of their processes. Combined results based on structural and temporal mining outcomes are intuitively auspicious and desirable if they yield a higher accuracy, higher confidence, and more meaningful insights. We still have to factor in that using both perspectives returns an ensemble technique. Ensemble data mining is often useful if we rely on a collection of multiple confident mining techniques. We apply ensembles that differ in their technical means but share a common output, e.g., determining the conformance score of a particular process case. Traditional process mining focuses mainly on structural anomalies. Duplicates, illicit orderings, or otherwise, invalid control flow are likely results. Using the time perspective for temporal conformance checking, we aim for unusual accelerations or delays that indicate a conformance issue. Developing a hybrid conformance checking approach is delicate, needs a suitable application with the known correlation between structural and temporal behaviors, and requires a meticulous balancing of all involved perspectives. In the best case, we balance in favor of only one of the perspectives, which yields a more inefficient hybrid approach than a structural process mining technique. However, in the worst case, anomalies might alert the applied techniques mutually exclusive, leading to a vast omission of critical conformance infringements.

1.8 Future Work

The majority of the discussed approaches regard the temporal perspective individually and independent of other perspectives. In the previous section, we recommended treating mining of temporal results and workflow aspects separately at first. Nevertheless, developing hybrid approaches is still rewarding and should be investigated under consideration of the previously mentioned points. Instead of building a two-factor ensemble, discovered structural information is useful to augment temporal process mining approaches. In most of the discussed approaches, we consider every relation of two sequential activities occurring in the process cases. A better solution is based on an initial determination of **milestone activities** for performance improvements. By relying on control-flow models, hub activities as effective centers between subprocesses provide meaningful candidates to establish the relation set and lead to decreased computation times.

Turning the tables, results from the temporal process mining domain improve structural discoveries. E.g., a detected temporal concept drift can **indicate** and finally lead to a **structural change**. In this thesis, we show how to detect drifts. The challenge here is not to blindly predict structural concept drifts but to classify temporal drifts before announcing false predictions. Not every detected concept drift impacts the workflow, and not every workflow change can be backtracked to a temporal concept drift. Techniques for specific concept drift detection are required to throw a bridge between temporal and structural perspectives.

Process mining has a strong focus on generalization. Emphasis is put on frequent process executions. In data mining, a novel research field emerged that considers rare patterns in sequence and transaction databases[57][29]. Especially in medical applications and in the security domain, rare patterns are an emerging research topic since detecting those is very rewarding in terms of life savings and fraud avoidance[26]. Considering collective anomalies, we slightly touched the surface of this topic. The **rare patterns** of processes are infrequent trace sequences with mutual behavior. We investigate the structural clusterings of non-conforming cases [47], and collective temporal anomalies [40]. By regarding only non-conforming cases, we already introduce a bias based on frequent behavior into the analysis. Temporal anomaly detection is a method to identify dense temporal patterns. Thus, it is limited to this perspective. A more in-depth analysis and research on rare process patterns seem to be rewarding.

Rare and frequent patterns are not confined to complete process cases and end-to-end process mining. Looking into partial cases, we discover more frequent subsequences of traces due to the more specific behavior of smaller episodes[32]. Instead of episodes, we can also pick various dimensions from a case representation and explore structures regarding a particular selection of features. In data mining, we apply the paradigm known as subspace clustering. Regarding density-based clustering, there are many approaches like the grid-based techniques CLIQUE[1] and its adaptive upgrade MAFIA [21]. Interesting candidate sets are generated using **subprocess clustering**. These sets yield more distinct and significant patterns. Especially in temporal anomaly detection, investigating the entire space of dimensions for activity relations is tedious. Various structures are neglected due to the curse of dimensionality. E.g., focusing on frequent subspaces, we also include patterns with correlations between episodes from process starts and ends.

Finally, we evaluate our approaches mainly on datasets with point-wise timestamps. Considering **intervals of activity lifecycle periods** instead of relations between two activities provides additional insights[11]. Previously, we only consider events without lifecycles. Thus our approaches rely on event completions since this allows more general evaluations as event logs usually contain at least one timestamp for each event. However, we lose accuracy due to neglected idle times or other types of delay. A process's performance is determined more precisely by taking temporal activity correlations into account, as shown in [31]. As process mining, while already quite mature, is a still-expanding research field, more applications provide their data logs, and more interval event logs get available.

Bibliography

[1] Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data* (1998), pp. 94–105.

[2] Agrawal, R., Imieliński, T., and Swami, A. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data* (1993), pp. 207–216.

[3] Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. Optics: ordering points to identify the clustering structure. *ACM Sigmod record 28*, 2 (1999), 49–60.

[4] Awad, A., Weidlich, M., and Sakr, S. Process mining over unordered event streams. In *2020 2nd International Conference on Process Mining (ICPM)* (2020), IEEE, pp. 81–88.

[5] Basile, F., Chiacchio, P., and Coppola, J. Model repair of time petri nets with temporal anomalies. *IFAC-PapersOnLine 48*, 7 (2015), 85–90.

[6] Böhmer, K., and Rinderle-Ma, S. Multi-perspective anomaly detection in business process execution events. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (2016), Springer, pp. 80–98.

[7] Böhmer, K., and Rinderle-Ma, S. Mining association rules for anomaly detection in dynamic process runtime behavior and explaining the root cause to users. *Information Systems 90* (2020), 101438.

[8] Ceci, M., Lanotte, P. F., Fumarola, F., Cavallo, D. P., and Malerba, D. Completion time and next activity prediction of processes using sequential pattern mining. In *International Conference on Discovery Science* (2014), Springer, pp. 49–61.

[9] Ceravolo, P., Tavares, G. M., Junior, S. B., and Damiani, E. Evaluation goals for online process mining: a concept drift perspective. *IEEE Transactions on Services Computing* (2020).

[10] Chaudhuri, S., and Dayal, U. An overview of data warehousing and olap technology. *ACM Sigmod record 26*, 1 (1997), 65–74.

[11] Chen, Y.-C., Peng, W.-C., and Lee, S.-Y. Mining temporal patterns in time interval-based data. *IEEE Transactions on Knowledge and Data Engineering 27*, 12 (2015), 3318–3331.

[12] Choueiri, A. C., Sato, D. M. V., Scalabrin, E. E., and Santos, E. A. P. An extended model for remaining time prediction in manufacturing systems using process mining. *Journal of Manufacturing Systems 56* (2020), 188–201.

[13] Cook, J. E., He, C., and Ma, C. Measuring behavioral correspondence to a timed concurrent model. In *Proceedings IEEE International Conference on Software Maintenance. ICSM 2001* (2001), IEEE, pp. 332–341.

[14] Crucitti, P., Latora, V., and Marchiori, M. Model for cascading failures in complex networks. *Physical Review E 69*, 4 (2004), 045104.

[15] Dumas, M., La Rosa, M., Mendling, J., and Reijers, H. A. *Business process management*. Springer, 2013.

[16] Eder, J., Panagos, E., Pozewaunig, H., and Rabinovich, M. Time management in workflow systems. In *BIS '99* (London, 1999), W. Abramowicz and M. E. Orlowska, Eds., Springer London, pp. 265–280.

[17] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (1996), vol. 96, pp. 226–231.

[18] Folino, F., Guarascio, M., and Pontieri, L. Discovering context-aware models for predicting business process performances. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (2012), Springer, pp. 287–304.

[19] Folino, F., Guarascio, M., and Pontieri, L. Discovering high-level performance models for ticket resolution processes. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (2013), Springer, pp. 275–282.

[20] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM computing surveys (CSUR) 46*, 4 (2014), 1–37.

[21] Goil, S., Nagesh, H., and Choudhary, A. Mafia: Efficient and scalable subspace clustering for very large data sets. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (1999), vol. 443, ACM, p. 452.

[22] Han, J., Pei, J., and Kamber, M. *Data mining: concepts and techniques*. Elsevier, 2011.

[23] Han, J., Pei, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., and Hsu, M. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern

growth. In *proceedings of the 17th international conference on data engineering* (2001), IEEE Washington, DC, USA, pp. 215–224.

[24] Hassani, M., Siccha, S., Richter, F., and Seidl, T. Efficient process discovery from event streams using sequential pattern mining. In *2015 IEEE symposium series on computational intelligence* (2015), IEEE, pp. 1366–1373.

[25] Heidegger, M. *History of the concept of time: Prolegomena*, vol. 717. Indiana University Press, 1992.

[26] Koh, Y. S., and Ravana, S. D. Unsupervised rare pattern mining: a survey. *ACM Transactions on Knowledge Discovery from Data (TKDD) 10*, 4 (2016), 1–29.

[27] Lefebvre, D. Detection of temporal anomalies for partially observed timed pns. *Mathematical Problems in Engineering 2017* (2017).

[28] Leitner, P., Wetzstein, B., Rosenberg, F., Michlmayr, A., Dustdar, S., and Leymann, F. Runtime prediction of service level agreement violations for composite services. In *Service-oriented computing. ICSOC/ServiceWave 2009 workshops* (2009), Springer, pp. 176–186.

[29] Lu, Y., Richter, F., and Seidl, T. Efficient infrequent itemset mining using depth-first and top-down lattice traversal. In *International Conference on Database Systems for Advanced Applications* (2018), Springer, pp. 908–915.

[30] Luna, J. M., Fournier-Viger, P., and Ventura, S. Frequent itemset mining: A 25 years review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 9*, 6 (2019), e1329.

[31] Maldonado, A., Sontheim, J., Richter, F., and Seidl, T. Performance skyline: Inferring process performance models from interval events. In *ICPM workshops proceedings 2020* (2020), Springer, p. tba.

[32] Mannila, H., Toivonen, H., and Verkamo, A. I. Discovery of frequent episodes in event sequences. *Data mining and knowledge discovery 1*, 3 (1997), 259–289.

[33] Mavroudopoulos, I., and Gounaris, A. Detecting temporal anomalies in business processes using distance-based methods. In *International Conference on Discovery Science* (2020), Springer, pp. 615–629.

[34] Monk, E., and Wagner, B. *Concepts in enterprise resource planning.* Cengage Learning, 2012.

[35] Navarin, N., Vincenzi, B., Polato, M., and Sperduti, A. Lstm networks for data-aware remaining time prediction of business process instances. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)* (2017), IEEE, pp. 1–7.

[36] Polato, M., Sperduti, A., Burattin, A., and de Leoni, M. Data-aware remaining time prediction of business process instances. In *2014 International Joint Conference on Neural Networks (IJCNN)* (2014), IEEE, pp. 816–823.

[37] Polato, M., Sperduti, A., Burattin, A., and de Leoni, M. Time and activity sequence prediction of business process instances. *Computing 100*, 9 (2018), 1005–1031.

[38] Richter, F., Lu, Y., Kazempour, D., and Seidl, T. Amtics: Aligning micro-clusters to identify cluster structures. In *International Conference on Database Systems for Advanced Applications* (2020), Springer, pp. 752–768.

[39] Richter, F., Lu, Y., Kazempour, D., and Seidl, T. “show me the crowds!” revealing cluster structures through amtics. *Data Science and Engineering* (2020), 1–15.

[40] Richter, F., Lu, Y., Zellner, L., Sontheim, J., and Seidl, T. Toad: Trace ordering for anomaly detection. In *2020 2nd International Conference on Process Mining (ICPM)* (2020), IEEE, pp. 169–176.

[41] Richter, F., Maldonado, A., Zellner, L., and Seidl, T. Otoso: Online trace ordering for structural overviews. In *ICPM workshops proceedings 2020* (2020), Springer, p. tba.

[42] Richter, F., and Seidl, T. Tesseract: time-drifts in event streams using series of evolving rolling averages of completion times. In *International Conference on Business Process Management* (2017), Springer, pp. 289–305.

[43] Richter, F., and Seidl, T. Looking into the tesseract: Time-drifts in event streams using series of evolving rolling averages of completion times. *Information Systems 84* (2019), 265–282.

[44] Richter, F., Sontheim, J., Zellner, L., and Seidl, T. TADE: stochastic conformance checking using temporal activity density estimation. In *Business Process Management - 18th International Conference, BPM 2020, Seville, Spain, September 13-18, 2020, Proceedings* (2020), D. Fahland, C. Ghidini, J. Becker, and M. Dumas, Eds., vol. 12168 of *Lecture Notes in Computer Science*, Springer, pp. 220–236.

[45] Richter, F., Wahl, F., Sydorova, A., and Seidl, T. k-process: Model-conformance-based clustering of process instances. In *LWDA* (2019), pp. 161–172.

[46] Richter, F., Zellner, L., Azaiz, I., Winkel, D., and Seidl, T. Liproma: Label-independent process matching. In *Business Process Management Workshops - BPM 2019 International Workshops, Vienna, Austria, September 1-6, 2019, Revised Selected Papers* (2019), C. D. Francescomarino, R. M. Dijkman, and U. Zdun, Eds., vol. 362 of *Lecture Notes in Business Information Processing*, Springer, pp. 186–198.

[47] Richter, F., Zellner, L., Sontheim, J., and Seidl, T. Model-aware clustering of non-conforming traces. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (2019), Springer, pp. 193–200.

[48] Rogge-Solti, A., and Kasneci, G. Temporal anomaly detection in business processes. In *International Conference on Business Process Management* (2014), Springer, pp. 234–249.

[49] Rogge-Solti, A., and Weske, M. Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In *International conference on service-oriented computing* (2013), Springer, pp. 389–403.

[50] Rogge-Solti, A., and Weske, M. Prediction of business process durations using non-markovian stochastic petri nets. *Information Systems 54* (2015), 1–14.

[51] Schubert, E., Sander, J., Ester, M., Kriegel, H. P., and Xu, X. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS) 42*, 3 (2017), 1–21.

[52] Senderovich, A., Weidlich, M., and Gal, A. Temporal network representation of event logs for improved performance modelling in business processes. In *International Conference on Business Process Management* (2017), Springer, pp. 3–21.

[53] Song, M., Günther, C. W., and Van der Aalst, W. M. Trace clustering in process mining. In *International conference on business process management* (2008), Springer, pp. 109–120.

[54] Sontheim, J., Richter, F., and Seidl, T. Temporal deviations on event sequences. In *LWDA* (2019), pp. 173–177.

[55] Srikant, R., and Agrawal, R. Mining sequential patterns: Generalizations and performance improvements. In *International Conference on Extending Database Technology* (1996), Springer, pp. 1–17.

[56] Stertz, F., Mangler, J., and Rinderle-Ma, S. Temporal conformance checking at runtime based on time-infused process models. *arXiv preprint arXiv:2008.07262* (2020).

[57] Szathmary, L., Napoli, A., and Valtchev, P. Towards rare itemset mining. In *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)* (2007), vol. 1, IEEE, pp. 305–312.

[58] Tattersall, I. Why was human evolution so rapid? In *Human Paleontology and Prehistory*. Springer, 2017, pp. 1–9.

[59] Van Der Aalst, W. Data science in action. In *Process mining*. Springer, 2016, pp. 3–23.

[60] Van Der Aalst, W. M. Process cubes: Slicing, dicing, rolling up and drilling down event data for process mining. In *Asia-Pacific conference on business process management* (2013), Springer, pp. 1–22.

[61] Van der Aalst, W. M., Pesic, M., and Song, M. Beyond process mining: from the past to present and future. In *International Conference on Advanced Information Systems Engineering* (2010), Springer, pp. 38–52.

[62] Van der Aalst, W. M., Schonenberg, M. H., and Song, M. Time prediction based on process mining. *Information systems 36*, 2 (2011), 450–475.

[63] van Dongen, B. F., Crooy, R. A., and van der Aalst, W. M. Cycle time prediction: When will this case finally be finished? In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (2008), Springer, pp. 319–336.

[64] Van Rijsbergen, C. J. Information retrieval. 2nd. newton, ma, 1979.

[65] Verenich, I., Dumas, M., Rosa, M. L., Maggi, F. M., and Teinemaa, I. Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. *ACM Transactions on Intelligent Systems and Technology (TIST) 10*, 4 (2019), 1–34.

[66] Weber, E. H. *De pulsu, resorptione, auditu et tactu: annotationes anatomicae et physiologicae, auctore.* prostat apud CF Koehler, 1834.

[67] Weijters, A., van Der Aalst, W. M., and De Medeiros, A. A. Process mining with the heuristics miner-algorithm. *Technische Universiteit Eindhoven, Tech. Rep. WP 166* (2006), 1–34.

[68] Yang, S., Zhou, M., Webman, R., Yang, J., Sarcevic, A., Marsic, I., and Burd, R. S. Duration-aware alignment of process traces. In *Industrial Conference on Data Mining* (2016), Springer, pp. 379–393.

[69] Zaki, M. J. Spade: An efficient algorithm for mining frequent sequences. *Machine learning 42*, 1-2 (2001), 31–60.

Chapter 2

Anomalous Event Detection

Publications and Declarations of Authorship

- Hassani, M., Siccha, S., Richter, F. and Seidl, T., 2015, December. Efficient process discovery from event streams using sequential pattern mining. In 2015 IEEE symposium series on computational intelligence (pp. 1366-1373). IEEE.
  DOI: 10.1109/SSCI.2015.195

  Declaration: *Marwan Hassani and I conceived the presented idea. I performed all computations and evaluations. I wrote the manuscript, except of the related work which has been done by Sergio Siccha. I regularly discussed this work with my supervisor Marwan Hassani, who supervised my master thesis. This work was not part of this examination, but has been conceived during a discussion. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

- Richter, F. and Seidl, T., 2017, September. TESSERACT: time-drifts in event streams using series of evolving rolling averages of completion times. In International Conference on Business Process Management (pp. 289-305). Springer, Cham.
  DOI: 10.1007/978-3-319-65000-5_17

  Declaration: *I conceived the presented idea and performed all computations and evaluations. I wrote the manuscript. I discussed the work with my supervisor Thomas Seidl, who assisted in polishing the final manuscript.*

- Richter, F. and Seidl, T., 2019. Looking into the TESSERACT: Time-drifts in event streams using series of evolving rolling averages of completion times. Information Systems 84, pp.265-282.
  DOI: 10.1016/j.is.2018.11.003

  Declaration: *This work is a journal extension of TESSERACT. I conceived the additional material and experiments, and performed all computations and evaluations. I wrote the extended manuscript.*

Chapter 3

Anomalous Trace Detection

Publications and Declarations of Authorship

- Sontheim, J., Richter, F. and Seidl, T., 2019. Temporal Deviations on Event Sequences. In LWDA (pp. 173-177).
  http://ceur-ws.org/Vol-2454/paper_73.pdf

  Declaration: *Janina Sontheim and I conceived the presented idea. I wrote the manuscript. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

- Richter, F., Lu Y., Zellner, L., Sontheim, J. and Seidl, T., 2020, October. TOAD: Trace Ordering for Anomaly Detection. In International Conference on Process Mining. IEEE.
  DOI: 10.1109/ICPM49681.2020.00033

  Declaration: *I conceived the presented idea and wrote the manuscript. I performed all computations and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

Chapter 4

Quickspotting Cluster Dynamics

Publications and Declarations of Authorship

- Richter, F., Lu, Y., Kazempour, D. and Seidl, T., 2020. AMTICS: Aligning Micro-clusters To Identify Cluster Structures. In International Conference on Database Systems for Advanced Applications (to be published). Springer, Cham.
  DOI: 10.1007/978-3-030-59410-7_52

  Declaration: *I conceived the presented idea and wrote the manuscript. I performed all computations and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

- Richter, F., Lu, Y., Kazempour, D. and Seidl, T., 2020. “Show Me the Crowds!” Revealing Cluster Structures Through AMTICS. Data Science and Engineering, pp.1-15.
  DOI: 10.1007/s41019-020-00137-x

  Declaration: *This work is a journal extension of AMTICS. I conceived the additional material and experiments, and performed all computations and evaluations. I wrote the extended manuscript. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

- Richter, F., Maldonado A., Zellner, L. and Seidl, T., 2021. OTOSO: Online Trace Ordering for Structural Overviews. Streaming Analytics for Process Mining.
  DOI: 10.1007/978-3-030-72693-5_17

  Declaration: *I conceived the presented idea and wrote the manuscript. I performed all computations and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

Chapter 5

Model-Conformance-based Trace Clustering

Publications and Declarations of Authorship

- Richter, F., Wahl, F., Sydorova, A. and Seidl, T., 2019. k-process: Model-Conformance-based Clustering of Process Instances. In LWDA (pp. 161-172).
  http://ceur-ws.org/Vol-2454/paper_46.pdf

  Declaration: *I conceived the presented idea and wrote the manuscript. Florian Wahl performed all computations and evaluations. Florian Wahl and I discussed very regularly about the approach and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

- Richter, F., Zellner, L., Sontheim, J. and Seidl, T., 2019, October. Model-Aware Clustering of Non-conforming Traces. In OTM Confederated International Conferences "On the Move to Meaningful Internet Systems" (pp. 193-200). Springer, Cham.
  DOI: 10.1007/978-3-030-33246-4_12

  Declaration: *I conceived the presented idea and wrote the manuscript. Ludwig Zellner and I performed all computations and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

Chapter 6

Time-based Conformance Matching

Publications and Declarations of Authorship

- Richter, F., Sontheim, J., Zellner, L. and Seidl, T., 2020, September. TADE: Stochastic Conformance Checking Using Temporal Activity Density Estimation. In International Conference on Business Process Management (pp. 220-236). Springer, Cham. DOI: 10.1007/978-3-030-58666-9_13

  Declaration: *I conceived the presented idea and wrote the manuscript. I performed all computations and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*

- Richter, F., Zellner, L., Azaiz, I., Winkel, D. and Seidl, T., 2019, September. LIProMa: Label-Independent Process Matching. In International Conference on Business Process Management (pp. 186-198). Springer, Cham.
  DOI: 10.1007/978-3-030-37453-2_16

  Declaration: *I conceived the presented idea and wrote the manuscript. I performed all computations and evaluations. I very regularly discussed with Imen Azaiz about the concepts and evaluations. All co-authors discussed the results periodically and assisted in polishing the final manuscript.*