applied sciences

MDPI

Review

Event Log Preprocessing for Process Mining: A Review

Heidy M. Marin-Castro [1,*] and Edgar Tello-Leal [2]

[1] CONACYT-Facultad de Ingeniería y Ciencias, Universidad Autónoma de Tamaulipas, Victoria 87000, Mexico
[2] Facultad de Ingeniería y Ciencias, Universidad Autónoma de Tamaulipas, Victoria 87000, Mexico; etello@docentes.uat.edu.mx
* Correspondence: hmarisol@docentes.uat.edu.mx

Abstract: Process Mining allows organizations to obtain actual business process models from event logs (discovery), to compare the event log or the resulting process model in the discovery task with the existing reference model of the same process (conformance), and to detect issues in the executed process to improve (enhancement). An essential element in the three tasks of process mining (discovery, conformance, and enhancement) is data cleaning, used to reduce the complexity inherent to real-world event data, to be easily interpreted, manipulated, and processed in process mining tasks. Thus, new techniques and algorithms for event data preprocessing have been of interest in the research community in business process. In this paper, we conduct a systematic literature review and provide, for the first time, a survey of relevant approaches of event data preprocessing for business process mining tasks. The aim of this work is to construct a categorization of techniques or methods related to event data preprocessing and to identify relevant challenges around these techniques. We present a quantitative and qualitative analysis of the most popular techniques for event log preprocessing. We also study and present findings about how a preprocessing technique can improve a process mining task. We also discuss the emerging future challenges in the domain of data preprocessing, in the context of process mining. The results of this study reveal that the preprocessing techniques in process mining have demonstrated a high impact on the performance of the process mining tasks. The data cleaning requirements are dependent on the characteristics of the event logs (voluminous, a high variability in the set of traces size, changes in the duration of the activities. In this scenario, most of the surveyed works use more than a single preprocessing technique to improve the quality of the event log. Trace-clustering and trace/event level filtering resulted in being the most commonly used preprocessing techniques due to easy of implementation, and they adequately manage noise and incompleteness in the event logs.

Keywords: process mining; data preprocessing; data quality; event log; noise event; data diversity

Citation: Marin-Castro, H.M.; Tello-Leal, E. Event Log Preprocessing for Process Mining: A Review. *Appl. Sci.* **2021**, *11*, 10556. https://doi.org/10.3390/app112210556

Academic Editor: Juan J. Rodríguez

Received: 23 September 2021
Accepted: 16 October 2021
Published: 10 November 2021

1. Introduction

Process mining is a relatively new study area that has gained significant attention among computer science and business process modeling communities [1]. It is a powerful tool for organizations to obtain actual models for better understanding of the real operation of their business processes and for better decision making. Process mining techniques allow automatic discovery, conformance, and improvement of process models implemented by organizations through the extraction of knowledge from event logs as well as from the available documentation of the process model [2]. In this context, an event log is a collection of time-stamped event records produced by the execution of a business process.

Considering that the event log is the main input for process mining techniques, the quality of this information has a great impact on the resulting model. An event log with low quality (missing, erroneous or noisy values, duplicates, etc.) can lead to a complex, unstructured (spaghetti-type), and difficult to interpret model (as shown in Figure 1a); or a model that does not reflect the real behavior of the business process. Therefore, event log data preprocessing is considered a task that can substantially improve the performance

of process mining. According with [3], in the big-data era, process mining tasks can be strongly limited by the quality of event data and processing times. This has placed greater attention on preprocessing tasks.

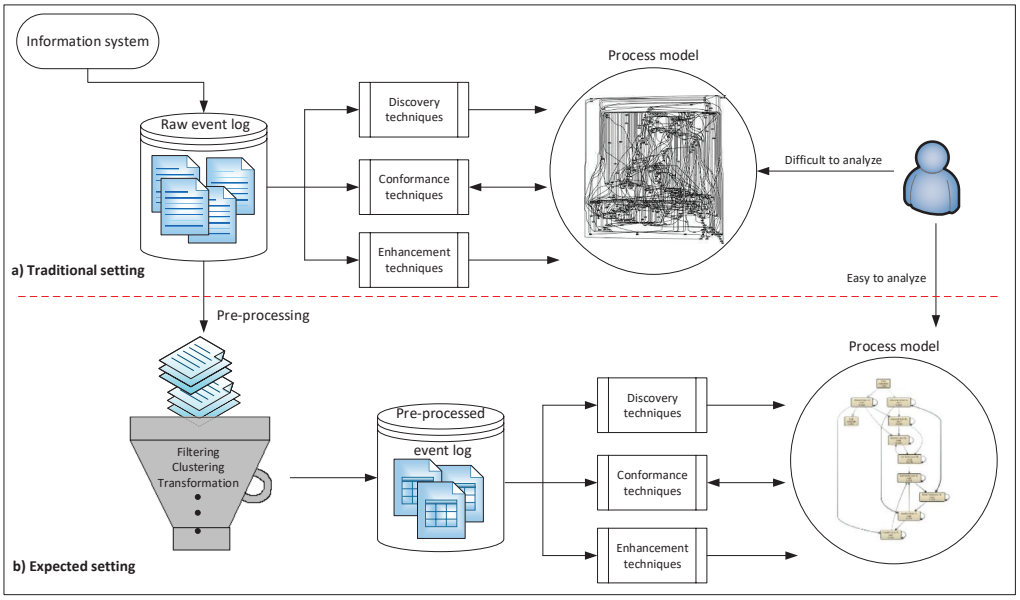

Figure 1. Overview of the traditional and expected process of applying data mining techniques using a raw event log and preprocessed event log, respectively.

Mans et al. [4] describe event log quality as a two-dimensional spectrum where the first dimension is concerned with the abstraction level (or granularity) of activities that are part of the process model. The second one is concerned with the accuracy of the timestamp (in terms of its granularity, directness of registration, and correctness) of the events registered in the log. Emamjome et al. [5] present the data quality problem from a theoretical approach addressing the root causes and the social, material, and individual factors that contribute to low quality data in event data, which would be overlooked by existing data cleaning methods. Other works [6,7] describe data quality as being a multi-dimensional concept associated with accuracy/correctness, completeness, unambiguity/understandability and timeliness. In this paper, we relate the event log quality with the identification, visualization, and correction, or elimination of incorrect, or noise, missing, duplicate, and irrelevant events. In real scenarios, some process mining tasks work under the assumption that behavior related to the execution of the running process is stored correctly within the event log, and that each instance of the process stored in the event log is already finished. However, real-world event logs contain noisy or corrupt data records, which can be generated by different factors: some traces are duplicated, incomplete, inconsistent, or reflect some other incorrect behavior. These problems can be caused by several factors, including errors in the data transmission, errors during storage, technology limitations, or transcript errors when events arrive in the wrong order. Sometimes, the noise may be associated to the presence of rare events due to handling of exceptional cases, incorrect recording of selected tasks in the execution of the process, or even for the incorrect assignment of timestamps. Constantly, it is difficult to distinguish between noise and infrequent correct behavior in an event log, resulting in a mined model with less fidelity to the real model. Different noise types in the event log impact negatively in quality issues, making that the process mining algorithm return complex, incomprehensible, or even inaccurate results. Therefore, to reduce these negative effects, event log preprocessing is a necessary task in the majority of process mining algorithms. The preprocessing of the event log attempts to detect and remove noise events, traces, or activities that contain such undesired behavior. This work provides a comprehensive review of the most representative techniques for preprocessing event logs, which is crucial for the performance of process mining tasks.

For the first time, we present a state-of-the-art review of the approaches for event log preprocessing that include techniques based on heuristics, pattern-based methods, trace clustering, and hybrid methods. We present an extensive analysis of the surveyed works, qualitatively and quantitatively.

Our review provides answers, in the broad sense, for three main questions: (1) how can the different techniques of event log preprocessing be grouped? (2) What problems exist around achieving data quality in the event log? and (3) How does one determine if a preprocessing technique could substantially improve a data mining task? For example, the grouping and identification of associated challenges to preprocessing techniques can serve to process mining implementers, to know the diverse types of available techniques, to provide them with more elements to select the most appropriate technique based on the underlying algorithms, the type of quality problems addressed, or particular issues in the application domain.

This research work has three main contributions:

1. We present, for the first time, a review of preprocessing techniques of event logs, also called data cleaning or data preparation techniques in the context of process mining.
2. We provide a grouping of preprocessing and repairing techniques of event logs, required to build more robust process models.
3. We present a study of relevant characteristics associated with preprocessing techniques used when making decisions about the use of a specific technique.

The remainder of this paper is organized as follows: Section 2 introduces the basic concepts related to event log preprocessing and to process mining. Section 3 presents the research methodology followed in this work to build this survey. Sections 3.2.1 and 3.2.2 present a proposal to group event log preprocessing techniques according to the approaches reported in the state-of-the-art. Section 3.3 outlines the tools used in each proposal submitted. Section 3.4 shows the representation schemes used for the manipulation and transformation of the event log. Section 3.5 presents the different problems identified in the event logs. Section 3.6 describes the tasks closely related to preprocessing. Section 3.7 identifies the attribute types to improve the quality of the event log. Section 4 provides insights on lessons learned and open problems. Finally, Section 5 concludes this work.

2. Preliminary Concepts

Process mining algorithms act over an event log—an event collection containing historical records from each business process instance. Each event produced during the execution of a business process instance (a case) corresponds to a trace. The set of all traces conform to the event log. This section presents some useful concepts for understanding the basis of event log preprocessing in the context of process mining.

Definition 1. *An event refers to a case, an activity, and a point in time. The event is characterized by a set of attributes such as, ID, timestamp, cost, resource, among others [8].*

Definition 2. *A trace can be seen as a case, i.e., a finite sequence of events $\sigma \in E^*$, such that each event appears only once [8].*

Definition 3. *An event log consists of a set of cases, and cases consist of events, such that each event appears, at most, once in the entire log [8].*

The events for a case are represented in the form of a trace, i.e., a sequence of unique events. Moreover, cases, such as events, can have attributes. The structure of an event log is made up of the following elements:

- An event log consists of cases.
- A case consists of events, such that each event relates to precisely one case.

- Events within a case are ordered.
- Events can have attributes. Examples of typical attributes are: activity name, time, costs, and resource.

Definition 4. *A Business process model is the graphical and analytic representation used to capture the behavior of an organization's business processes.*

A business process model is usually expressed through different graphic methods or notation languages, such as the flowchart, UML, workflows, Petri nets, BPMN, among others.

In the context of event log preprocessing, it is essential to identify the issues closely related to the quality of the data recorded in the event log. Therefore, some of the data quality issues that commonly occur in event logs are described below.

Noise/anomalous data: this problem corresponds to the scenario where the data in the event log contain errors, or meaningless data that deviate from the expected behavior. Incorrect or noise data may be the result from inconsistency or discrepancy in naming conventions or data codes used, or inconsistent formats for input fields, such as timestamps. Hence it is necessary to use some techniques to eliminate or replace the noisy data.

Missing data: this problem can occur when different information can be missing in the event log, although it should be registered on a mandatory basis. This occurs, for example, when an attribute in an event is missing due to problems related to the sending, registration, or storage of events from an information system.

Irrelevant data: in this scenario, there may be event records that are irrelevant for the analysis of the model under study, but from these, it is possible to derive the record of a relevant event through some transformation and filtering processes.

Duplicated data: this problem is present when the same event is recorded in the event log more than once, by the same resource and with the same timestamp. Likewise, the problem may arise when an activity is registered more than once by the information system, sometimes causing the process model to become a complex model.

Data diversity: this situation is present when the information system is very general and allows the diverse registration of events at different levels of granularity, which makes process models incomprehensible and difficult to represent.

Many of the issues of data quality previously mentioned have been addressed in the surveyed preprocessing techniques in this work.

3. Research Methodology

This section describes the methodology for the literature review presented in this work and the inclusion and exclusion criteria established for the selection of the surveyed works.

3.1. Systematic Review Process

The search and selection strategy of research works for this review was conducted in two stages. The first stage consisted of recovering the related works from three popular electronic libraries, including IEEE Xplore, Springer Link, and Science Direct, oriented on various disciplines, including process mining. Specifically, we collected papers since 2005 (period from which automatic algorithms for mining processes began to be proposed, such as the alpha algorithm) using the following terms *"refining, repairing, cleaning, refinement, filtering, clustering, preprocessing, ordered, aligning, abstraction, anomalous detection, infrequent behavior, noisy, imperfection, traces, event log, process mining"* identified in their title or abstract. These terms were combined to form search strings that serve as input queries to the three digital libraries (see Table 1). Given the generality of the articles retrieved from the selected digital libraries, in the second stage a search and selection strategy was applied in the inclusion/exclusion criteria, to decide which articles would be included in the final review.

3.1.1. Inclusion Criteria

Below are the inclusion criteria that were used to select the works analyzed and discussed in this review.

1. Research works written in English.
2. Research works published in journals, conferences, or theses.
3. Works published from at least 2005.
4. Works explicitly addressing the preprocessing or cleaning of event logs.

Table 1. Query details and results (number or retrieved papers) from the selected digital libraries.

Search String	IEEE Xplore	Springer Link	Science Direct
(“filtering” OR “cleaning” OR “repairing” OR “clustering” OR “refinement” OR "preprocessing") “event log” ”process mining”	110	32,440	310
(“filtering” OR “cleaning” OR “repairing” OR “clustering” OR “refinement” OR "preprocessing") “trace” ”process mining”	227	63,897	401
(“ordered” OR “aligning”) “event log” “process mining”	139	41,385	213
(“anomalous detection” OR “infrequent behavior” OR “noisy” OR “imperfection”) “event log” “process mining”	16	1813	87
(“anomalous detection” OR “infrequent behavior” OR “noisy” OR “imperfection”) “trace” “process mining”	24	2106	95

3.1.2. Exclusion Criteria

The following are the criteria to discard research works that are not of interest for this revision.

1. Works that are not related to process mining.
2. Works that do not focus on specific domains in the field of process mining (industry, manufacturing); that is, ad hoc techniques for a given domain.
3. Works that do not include evaluation and experimental results.

After filtering by the queried topic and removing the duplicated retrieved papers, a total of 95 papers were obtained and analyzed, considering the inclusion criteria. On this set, and after applying the exclusion criteria, the result was a set of 70 papers. All of these are included in our qualitative analysis.

Figure 2 shows the distribution of the selected works (in %), from 2006 to 2020, based on the year of publication (during a period of three years). Although Figure 2 reveals a slow growing tendency, it is difficult to determine if that tendency will be kept.

As shown in Figure 3, 25% of the selected works were published in journals, 70% were reported at international conferences, and the rest are theses. Works formerly reported (2006–2008) have had great influence in the community, as Figure 4 reveals a total of around 2400 cites. Moreover, most recent works (2017–2020) reveal more than 350 cites.

Figure 5 shows a network of closely related terms that describe the different addressed topics of event log preprocessing. This network was formed from the relationships identified between concepts included in the abstract of the surveyed works. The color intensity in the network nodes refers to more general or abstract terms that contain other more specific terms, all related to describing the handling of preprocessing in process mining. This network of terms could be useful for scientists to understand and organize the diversity of existing preprocessing techniques.

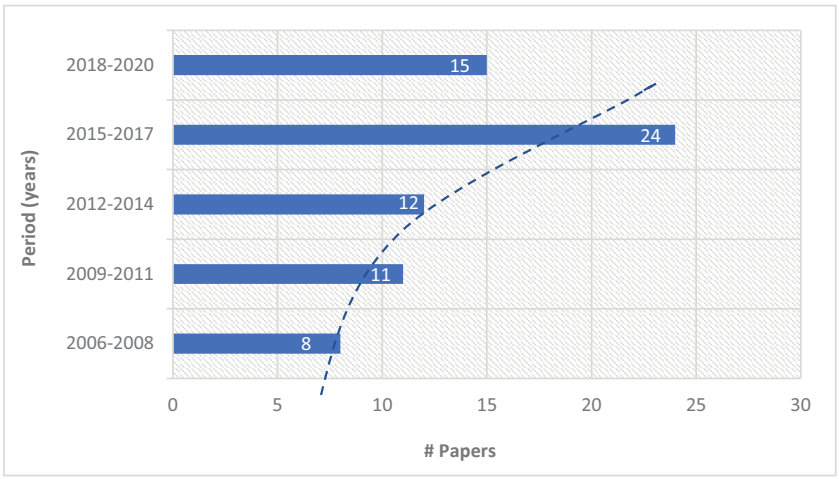

Figure 2. Publications, each three years of the surveyed papers.

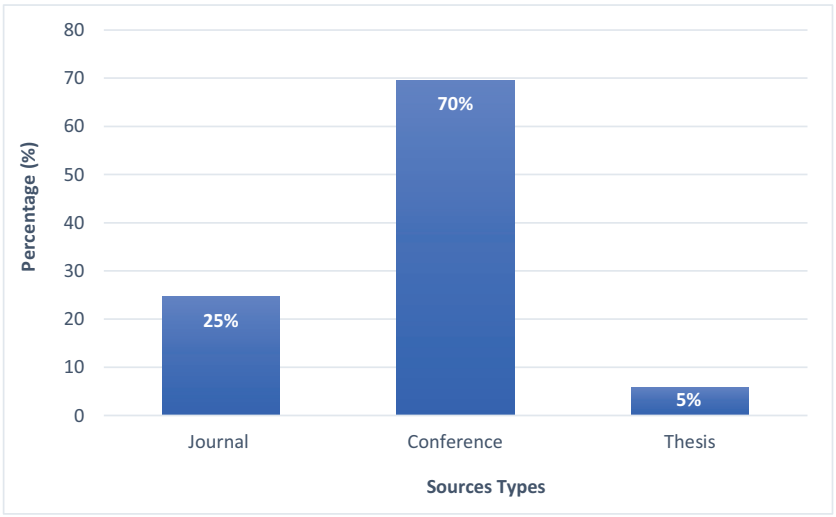

Figure 3. Percentage of publications associated with different surveyed sources in this survey.

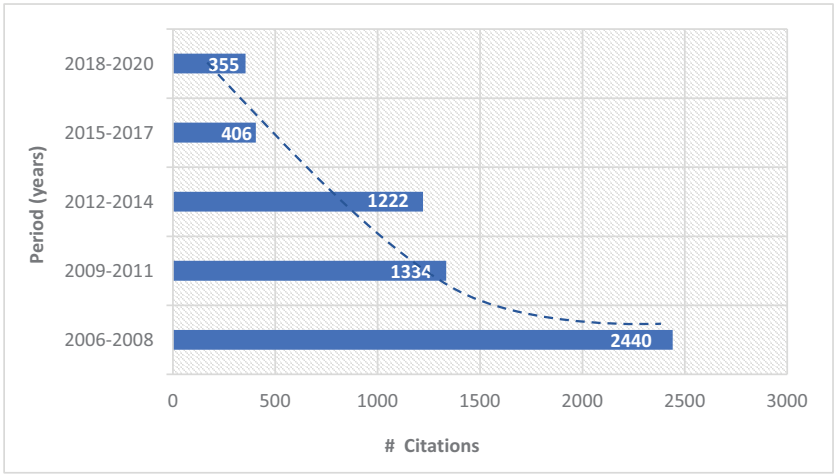

Figure 4. Citations, each three years of the surveyed papers.

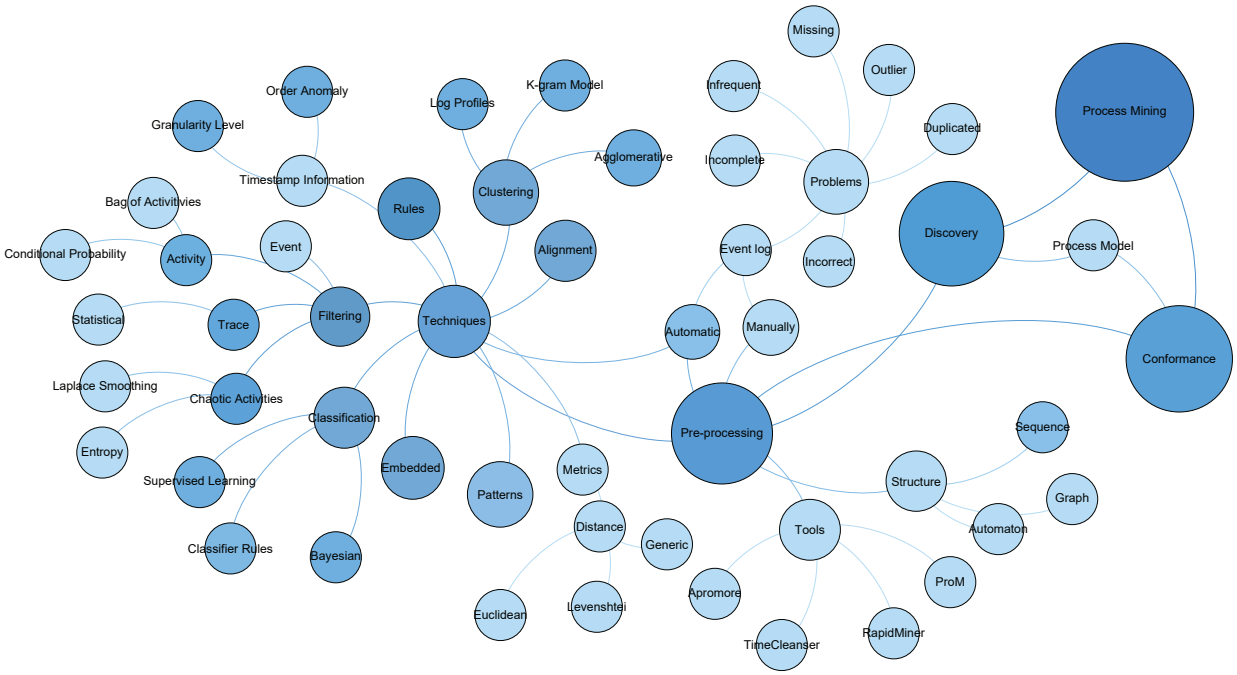

Figure 5. Summary of different closely related terms and their relations in the data preprocessing domain in process mining.

During the literature review, a content study was performed. In this study, we identified and classified the common and relevant characteristics found in the surveyed papers. Table 2 outlines a general view and a summary of the most significant characteristics (C1—techniques, C2—tools, C3—representation schemes, C4—imperfection types, C5—related tasks, and C6—types of information), which are described in greater detail in the next sections.

Table 2. Main characteristics in the reviewed studies.

ID	Characteristic	Description
[C1]	Techniques	Two main families of techniques: (1) transformation techniques and (2) detection and visualization techniques
[C2]	Tools	ProM, Disco, RapidProM, Celonis, Apromore, RapidMiner, Java application, preprocessing framework
[C3]	Representation schemes	Sequences of events/traces or vectors, graphs, automatons
[C4]	Imperfection types	Form-based event capture, inadvertent time travel, unanchored event, scattered event, elusive case, scattered case, collateral events, polluted label, distorted label, synonymous labels, homonymous label, timestamp granularity, unusual temporal ordering
[C5]	Related tasks	Two types: event abstraction and alignment
[C6]	Types of information	Event label, timestamp, ID, cost, resource, additional event payload

3.2. C1. Techniques

Is there a way of grouping event log preprocessing techniques?

Different criteria might lead to different taxonomies of data preprocessing techniques in the context of process mining. From the surveyed works, we organize the existing event log preprocessing techniques, in two main groups: transformation techniques and detection–visualization techniques. The main classification criterion is the approach followed by the preprocessing techniques to clean the data, which includes identification, isolation, and reparation of errors. Figure 6 schematically shows a possible taxonomy for the surveyed works. The proposed taxonomy organizes the diversity of existing preprocessing techniques and helps identify characteristics that they may have in common. Our grouping also serves to identify in which data quality issues that certain types of techniques are more suitable to use. The first category consists of techniques that perform transfor-

mations in the event log in order to correct the imperfect behaviors (missing, irrelevant, duplicate data, etc.), before applying a process mining algorithm. The second category is comprised of techniques to detect or diagnose imperfections in an event log. While the second category of techniques only detect potential problems related to data quality in the event log, the techniques in the first category directly correct the imperfections found in the event log.

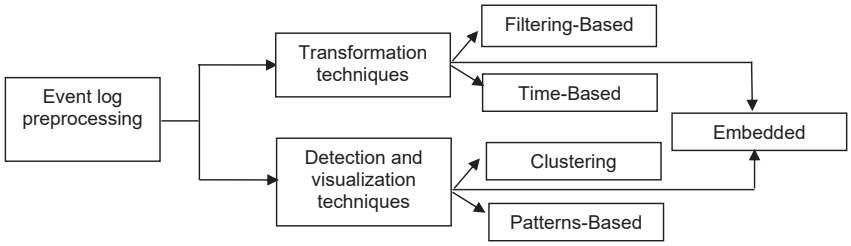

Figure 6. Proposed grouping for data preprocessing in process mining divided into two main families, transformation, and detection–visualization techniques.

3.2.1. Transformation Techniques

Transformation techniques carry out operations and actions to mark changes in the original structure of the raw event log in order to improve the quality of the log. Within this group, there are two main approaches: filtering and time-based techniques. On the one hand, filtering techniques aim to determine the likelihood of the occurrence of events or traces based on its surrounding behavior. The events or traces with less frequency of occurrence are removed from the original event log. Filtering techniques are focused on removing logging mistakes to prevent their spreading to the process models. On the other hand, the objective of time-based techniques is to maintain and correct the order of the events recorded in the log from the timestamp information.

Filtering techniques fundamentally address the search and elimination of noise/anomalous events or traces with missing values. Their main characteristics involve the filtering of atypical behavior identified in the event log that may affect the performance of future process mining tasks. These techniques model the frequently occurring contexts of activities and filter out the contexts of events that occur infrequently in the log.

There are several works [9–15] reported in the literature that propose the development of filtering techniques. Conforti et al. [10] presented a technique that relies on the identification of anomalies in a log automaton. First, the technique builds an abstraction of the process behavior recorded in the log as an automaton (a directed graph). This automaton captures the direct follow dependencies between events in the log. Infrequent transitions are subsequently removed using an alignment-based replay technique while minimizing the number of events removed from the log.

van Zelst et al. [11] proposed an online/real-time event stream filter designed to detect and remove spurious events from event streams. The main idea of this approach is that dominant behavior attains higher occurrence probabilities within the automaton compared to spurious behavior. This filter was implemented as an open-source plugin for both ProM [16] and RapidProM [17] tools.

Wang et al. [9] presented the study of techniques for recovering missing events; thus, providing a set of candidates of more complete provenance. The authors used a backtracking idea to reduce the redundant sequences associated to parallel events. A branching framework was then introduced, where each branch could apply the backtracking directly. The authors constructed a branching index and developed reachability checking and lower bounds of recovery distances to further accelerate the computation.

Niek et al. [15] proposed four novel techniques for filtering out chaotic activities, which are defined as activities that do not have clear positions in the event sequence of the process model, for which the probability to occur does not change (or changes little)

as an effect of occurrences of other activities, i.e., the chaotic activities are not part of the process flow.

Within preprocessing approaches based on event-level filtering, [12–14] used trace sequences as a structure for managing the event log. This structure allows, in most of these works, the ordering and calculation of the frequency of occurrence of events for the identification of noise/anomalous behavior in the event log.

Other works, such as in [18–21], present algorithms for detection and removal of anomalous traces of process-aware systems, where an anomalous trace can be defined as a trace in the event log that has a conformance value below a threshold provided as input for the algorithm. That is, anomalous traces, once discovered, must be analyzed to find out if they are incorrect executions or if they are acceptable but uncommon executions.

Cheng and Kumar [22] aimed to build a classifier on a subset of the log, and apply the classifier rules to remove noisy traces from the log. They presented two proposals; the first one to generate noisy logs from reference process models, and to mine process models by applying process mining algorithms to both the noisy log and the sanitized version of the same log, then comparing the discovered models with the original reference model. The second proposal consisted of comparing the models obtained before and after sanitizing the log using structural and behavior metrics.

Mohammadreza et al. [23] proposed a filtering approach based on conditional probabilities between sequences of activities. Their approach estimates the conditional probability of occurrence of an activity based on the number of its preceding activities. If this probability is lower than a given threshold, the activity is considered as an outlier. The authors considered both noise and infrequent behavior as outliers. Furthermore, they used a conditional occurrence probability matrix (COP-Matrix) for storing dependencies between current activities and previously occurred activities at larger distances, i.e., subsequences of increasing length. Other techniques to filter anomalous events or traces are presented in [19,20,22,24–27].

Time-based techniques are other types of transformation techniques for data preprocessing in event logs. A wide variety of research works on event log preprocessing have focused on data quality issues related to timestamp information and their impacts on process mining [12,28]. Incorrect ordering of events can have adverse effects on the outcomes of process mining analysis. According to the surveyed works, time-based techniques have shown better results in data preprocessing. In [12,29], the authors established that one of the most latent and frequent problems in the event log is the one associated with anomalies related to the diversity of data (level of granularity) and the order in which the events are recorded in the logs. Therefore, strategies based on timestamp information are of great interest in the state-of-the-art.

Dixit et al. [12] presented an iterative approach to address event order imperfection by interactively injecting domain knowledge directly into the event log as well as by analyzing the impact of the repaired log. This approach is based on the identification of three classes of timestamp-based indicators to detect ordering related problems in an event log to pinpoint those activities that might be incorrectly ordered, and an approach for repairing identified issues using domain knowledge.

Hsu et al. [30] proposed a k-nearest neighbor method for systematically detecting irregular process instances using a set of activity-level durations, namely execution, transmission, queue, and procrastination durations. Activity-level duration is the amount of time required to complete an activity and contextual information, such as employee information and customer transactions extracted from the ERP system of a medium-sized logistics company. The distances between instances were calculated using the differences between adjusted durations.

Tax et al. [31] proposed a framework for the automated generation of label refinements based on the time attribute of events, allowing to distinguish behaviorally different instances of the same event type based on their time attributes. The events generated by one sensor were clustered using a mixture model consisting of components of the

von Mises distribution, which is the circular equivalent of the normal distribution. Four strategies were applied for multiple label refinements on three event logs from the human behavior domain.

Song et al. [28] proposed an approach based on the minimum change principle to repair timestamps that do not conform to temporal constraints, e.g., to find a repair that is as close as possible to the original observation. The problem is tackled by identifying a concise set of promising candidates using an algorithm for computing the optimal repair from the generated candidates, and a heuristic approximation by selecting repairs from the candidates.

Rogge-Solti et al. [32] presented a method to repair the timed event logs by combining stochastic Petri nets, alignments, and Bayesian networks. The method decomposes the problem into two sub-problems: (a) repairing the time and (b) repairing the structure for each trace. This work takes all of the observed data into account and gets efficient estimations for the activity durations and path probabilities.

Fischer et al. [33] proposed an approach for detecting and quantifying timestamp imperfections in event logs based on 15 quality metrics structured along four data quality dimensions and log levels.

3.2.2. Detection–Visualization Techniques

Detection–visualization techniques aim to identify, group, and isolate those events or traces that can generate problems in the quality of the event log. Within this group, two approaches are identified: clustering and pattern-based techniques. Clustering techniques divide the event log into several subsets, facilitating the understanding and analysis of each member of the subsets. Then, the next step is the identification of noise/anomalous elements within the analyzed subsets. Clustering is one of the techniques most used for data preprocessing in process mining, which has been mostly used for the identification of quality issues associated with noisy values, as well as data diversity. From the formation of similar clusters, it is possible to identify imperfection patterns related to noisy data in the different attributes of the event logs.

Several techniques have been proposed in the last decade for trace clustering. They can be divided into three approaches: vector space approaches [34–37], context aware approaches [38–42], and model-based approaches [43–48]. Most of the clustering algorithms aforementioned consider only the event log as input, and use different internal representations for producing the clusters. Traditionally, these algorithms have been applied without taking into consideration the availability of a process model. In contrast, in recent works [49,50], a different view on traces clustering of an event log is presented. The authors assume that a process model exists and it is used to build simpler groupings of homogeneous traces.

Table 3 summarizes the most relevant characteristics of the surveyed works of clustering techniques.

Table 3. Summary of event log preprocessing techniques using the clustering approach.

Year	Authors	Ref	Model	Approach	Algorithms
2019	Boltenhagen et al.	[50]	Framework for trace clustering of process behavior	Based on generalized alignment	Trace clustering ATC, APOTC, or AMSTC
2019	Xu and Liu	[37]	Trace clustering using log profiles	Based on trace profiles and missing trace profiles	Self-Organizing Map (SOM)
2017	Chatain et al.	[49]	Technique trace clustering	Based on the concept of multi-alignments, which groups log traces according to representative full runs of a given model, considering the problem of alignment	A pseudo-Boolean solver Min- isat+

Table 3. *Cont.*

Year	Authors	Ref	Model	Approach	Algorithms
2017	Yaguang et al.	[42]	Compound trace clustering	Convert the trace clustering problem based on notion of similarity trace into a clustering problem guided by the complexity of the sub-process modes derived from sub-logs	(1) context aware trace clustering technique (GED); (2) sequence clustering technique (SCT); (3) flexible heuristic miner (FHM) to discover process models (4) HIF algorithm to locate behavioral patterns recorded in the event log
2016	Evermann et al.	[36]	K-means trace clustering	Based on local alignment of sequences and subsequent multidimensional scaling	Smith–Waterman–Gotoh algorithm for sequence alignment, k-means clustering
2016	Nguyen et al.	[47]	Hierarchical trace clustering	Using the process traces representation to reduce the high dimensionality of event logs	(1) Greedy approximation algorithm based on extensible heterogeneous information networks (HINs). (2) Heuristics miner
2015	B. Hompes et al.	[41]	Trace clustering	Finding variations and deviations of a process based on a set of selected perspectives	Markov cluster (MCL) algorithm
2013	De Weerdt et al.	[46]	Active trace clustering	Based on a top-down greedy approach inspired in active learning to solve the problem of finding an optimal distribution of execution traces over a given number of clusters	(1) A selective sampling strategy; (2) Heuristics miner
2012	R. Jagadeesh et al.	[40]	Trace clustering	A context-aware approach by defining process-centric feature and syntactic methods based on edit distance	Agglomerative hierarchical clustering algorithm
2011	Folino et al.	[48]	Markov, k-means and agglomerative hierarchical aware clustering	Based on the similarity criterion among the traces via a special kind of frequent structural patterns, which are preliminary discovered as an evidence of "normal" behavior	(1) Decision-tree algorithm; (2). OASC: an algorithm for detecting outliers in a process log; (3) LearnDADT: an algorithm for inducing a DADT model
2011	Wang et al.	[39]	Suffix tree clustering	A context aware approach for identifying patterns that occur in traces. It uses a suffix-tree based approach to categorize transformed traces into clusters	(1) An equivalent of a single-link algorithm to group base clusters into end clusters; (2) Alpha++ mining algorithm to generate process models of clusters
2010	Chandra Bose and van der Aalst	[35]	Agglomerative hierarchical trace clustering	Based on multiple feature sets for trace clustering considering sub-sequences of activities conserved across multiple traces	(1) Ukkonen algorithm [51] for the construction of suffix-trees in linear-time; (2) Heuristics Miner algorithm to evaluate the goodness of clusters
2009	Jagadeesh and van der Aalst	[38]	Agglomerative hierarchical trace clustering	Based on: (a) bag-of-activities, (b) k-gram model, (c) Levenshtein distance, and (d) generic edit distance	(1) Algorithm to maximize the score of two sequences based on the similarity; (2) algorithm to generates the scores for the insertion of activities; and (3) algorithm to evaluate the significance of clusters
2008	Minseok et al.	[34]	Trace clustering using Log profiles	Based on the divide and conquer approach in which profiles measure a number of features for each case	(1) k-Means; (2) quality threshold (QT); (3) agglomerative hierarchical clustering, (4) self-organizing maps (SOM)
2008	A. K. A. de Medeiros et al.	[45]	Trace clustering algorithm that avoids over-generalization	Iteratively splitting the log in clusters	K-means clustering algorithm
2007	Ferreira et al.	[44]	Sequence clustering	This approach is useful in new scenarios, where the business process analyst may not be familiar with, or where the potential for process mining is yet uncertain	Sequence clustering algorithm based on first-order Markov chains, expectation-maximization (EM) algorithm
2006	Greco et al.	[43]	Hierarchical trace clustering	Based on an iterative hierarchical refinement of a disjunctive schema	A greedy strategy

Within the detection–visualization techniques, some of them perform the preparation of event logs from the pattern identification based on the definition and application of heuristic rules. These rules are identified from observed behaviors or acquired experiences by expert analysts in process mining from the study of different event logs in different domains. Many of the pattern-based techniques state that the event log is not completely correct if a given pattern is not detected in the log [29]. These techniques usually work in conjunction with clustering and abstraction or alignment techniques; thus, allowing the identification of patterns related to noisy data or data diversity.

Suriadi et al. [29] propose determining event log quality by the description of a collection of eleven log imperfection patterns obtained from their experiences in preparing event logs. The definition of pattern is given as the abstraction from a concrete form, which keeps recurring in specific non-arbitrary contexts.

Ghionna et al. [52] describe an approach that combines the discovery of frequent execution patterns with a cluster based anomaly detection procedure. Special algorithms are used for decreasing the counting of spurious activities and for coding a method that simultaneously clusters a log and its associated S-patterns, respectively (patterns and clustering).

WoMine-i [53] extracts, infrequently, components in the logs from the model specification (tasks sequences, selections, parallels, loops, etc.). WoMine-i performs an a priori search starting with the minimal patterns and reduces the search space by pruning the infrequent patterns.

Jagadeesh et al. [54] propose an iterative method for transforming traces that identify the looping constructs and sub-processes and replace the repeat occurrences by an abstracted entity. Other pattern-based approaches are presented in [14,35,39,48,54]. Additionally, some process mining algorithms [55–61] incorporate mechanisms of event log preprocessing (embedded techniques) as part of their approach. These algorithms implicitly attempt to detect noise traces, hidden tasks, duplicate activities in the event log, which can sometimes be attributed to event ordering imperfections. However, the decisions and detections made during the execution of some process mining tasks (discovery, conformance, or enhancement) are implicitly incorporated in the discovered process model. In this case, embedded preprocessing techniques are able to exploit their coupling to the discovery technique, allowing each step or iteration to verify and validate if the built process model is a solid model. This is revealed from some works [60,62], where it is ensured that, from the identification of noisy data, as well as from the flexible configuration of parameters in the preprocessing techniques, it is possible to build more solid and robust models.

Table 4 presents a general summary of some of the most popular event log preprocessing techniques previously discussed. In that table, we provide a notation to refer to a particular technique being used: $A1$ (event/trace level filtering), $A2$ (clustering), $A3$ (pattern-based techniques), $A4$ (Embedded techniques), $A5$ (time-based techniques), $B1$ (alignment), $B2$ (abstraction). Table 4 also shows the particular task (discovery-D, conformance-C or enhance-E) that is intended to be improved by including a preprocessing technique in that same order. In the table are also shown the main problems identified in the event log, such as missing data (*mis*), noise data (*noi*), diversity data (*div*), irrelevant data (*irr*), and duplicate data (dup). From Table 4, we can conclude that the trace clustering technique, and event/trace filtering are the two most frequently used techniques for the preprocessing task in process mining. Time-based preprocessing techniques have recently shown promising results in data preprocessing through the study, correction, and elimination of data associated with the timestamp attribute. Moreover, the table reveals that a vast majority of preprocessing techniques have been designed to improve the process model discovery, in order to improve the quality of the discovered models, reducing the complexity of the model through the management of clean data registered in the event log. Furthermore, about 60% of the studied techniques are available in process mining tools, such as ProM tool and a small percentage corresponds to individual applications that incorporate preprocessing techniques independently. Finally, Table 4 shows the two

most frequent problems in event logs, the presence of noise and the data diversity or granularity level.

Table 4. Summary of the reviewed data preprocessing techniques in the context of process mining.

Ref	Tech	Tools	Task			Representation	Problem				
			D	*C*	*E*		*mis*	*noi*	*div*	*irr*	*dup*
[43], 2006	A2	ProM	✓	✓		Dependency graph			✓		
[44], 2007	A2	Application				Sequences of activities			✓		
[55], 2007	A1,B2	Fuzzy Miner in ProM	✓			Graphs				✓	
[52], 2008	A3,A1,A2	Prototype system in Java	✓	✓	✓	Graphs		✓			
[34], 2008	A2	ProM	✓			Self-Organizing map (SOM)			✓		
[56], 2008	A4	–	✓			SWF-nets					✓
[18], 2008	A1	–	✓			Sequences of graphs and traces		✓			
[45], 2008	A2	ProM	✓			Sequences of events			✓		
[24], 2009	A1	ProM	✓			Sequences of traces		✓			
[54], 2009	A3,B2	ProM	✓			Sets of traces				✓	
[38], 2009	A2	–	✓			K-gram model			✓		
[57], 2009	A4	Enhanced-WFMiner	✓			Workflow schema	✓	✓			✓
[35], 2010	A2,A3	–	✓			Context-aware feature sets			✓		
[63], 2010	B1	ProM	✓	✓		Sets of traces	✓	✓			
[19], 2010	A1	ProM	✓			Sequences of traces		✓			
[64], 2010	B2,A2	ProM	✓			Graphs Simple Precedence Diagram			✓		
[65],2010	B2,A2	Stream Scope in Prom	✓			Sequences of events			✓		
[58], 2011	A4	ProM	✓			Augmented-C-nets		✓			
[48], 2011	A2,A3	Java prototype		✓		Set of traces		✓			
[39], 2012	A2,A3	ProM	✓			Suffix-Trees			✓		✓
[66], 2012	B1	ProM	✓	✓		Sequences of traces		✓	✓		
[40], 2012	A2,A1,B2	ProM	✓	✓		Sequences of traces	✓		✓		
[20], 2012	A1	ProM	✓	✓			✓	✓			
[67], 2012	B1	ProM		✓		Constraint Automaton		✓			
[46], 2013	A2	ActiTraC in ProM 6	✓			Sequences of traces		✓			
[68], 2013	B1,A5	ProM		✓		Bayesian networks	✓				
[59], 2013	A4	ProM	✓			Graphs		✓		✓	
[32], 2013	A1,B1			✓		Sequences of traces	✓				
[21], 2013	A1,B1	Algorithms in ProM	✓	✓		Sequences of traces		✓			
[69], 2013	B2	ProM	✓	✓	✓	Sequence of activity instances			✓		
[60], 2014	A4	IMi in ProM	✓			Process trees		✓	✓		
[22], 2015	A1	Weka tool for creating rules	✓			Sequences of traces		✓			
[70], 2015	B1	ProM		✓		Partially ordered traces		✓			✓
[71], 2015	B1,A1	-		✓	✓	Workflow nets	✓				
[41], 2015	A2	ProM				Sequences of traces		✓			
[36], 2016	A2,B1	Java application (AlignCluster) and ProM	✓			Sequences of traces			✓		
[47], 2016	A2	–	✓			Heterogeneous graph		✓	✓		
[9], 2016	A1,B1	Java programs	✓	✓		Sequences of events	✓				

Table 4. *Cont.*

Ref	Tech	Tools	Task			Representation	Problem				
			D	*C*	*E*		*mis*	*noi*	*div*	*irr*	*dup*
[72], 2016	A2,A1	TraceMatching tool in ProM	✓			Sequences of traces		✓	✓		
[28], 2016	A5	-			✓	Temporal networks		✓			
[73], 2016	B2,A2	-	✓			Casual activity graphs				✓	
[25], 2016	A1	Prototypical implementation				Likelihood graphs		✓	✓	✓	
[74], 2016	B2,B1,A3	ProM	✓			Sequence of events		✓	✓		
[75], 2016	B1	General framework and plugin Effa of ProM	✓	✓	✓	Sequence of event and graphs	✓	✓			✓
[10], 2017	A1	ProM	✓			Log automaton		✓		✓	
[29], 2017	A3	Disco	✓			-	✓	✓	✓	✓	✓
[76],2017	B2,A3,A2	-	✓			Sequence of events			✓		
[23], 2017	A1	ProM,RapidProM	✓	✓		Sequences of activities		✓		✓	
[30], 2017	A5	–			✓	Acyclic graphs		✓			✓
[31], 2017	B2,A5	Framework	✓			Sequences of traces				✓	
[53], 2017	A3	WoMine-i app			✓	Causal nets					
[61], 2017	A4	DHM in ProM	✓			Causal nets		✓			
[42], 2017	A2	(CTC) Compound Trace Clustering	✓		✓	Sequences of traces			✓		
[49], 2017	A2,B1	tool DarkSider		✓	✓	Process model rendered traces		✓	✓		
[62], 2017	A4	Fodina	✓			Causal nets		✓			✓
[14], 2018	A1	ProM, RapidProM	✓			Sequences of activities		✓			
[12], 2018	A1,A5	ProM	✓			Sequences of activities					✓
[77], 2018	B2	ProM	✓	✓		Feature vectors				✓	
[11], 2018	A1	ProM and RapidProM	✓	✓		Probabilistic automata		✓			
[13], 2018	A1	ProM and RapidProM	✓			Sequences of activities		✓			
[15], 2019	A1	RapidProM	✓			Sequences of activities		✓			
[37], 2019	A2	Framework for missing event log	✓			Profiles-trace vector	✓				
[50], 2019	A2,B1	Prototype tool	✓	✓		Sequence of traces					
[26], 2019	A1	Java application	✓			Without loop traces	✓				✓
[27], 2019	A1	Prom, RapidProM	✓			Sequences of activities		✓		✓	
[78], 2019	B2,A5	ProM	✓			Sequences of events		✓			✓
[79], 2020	A1	–	✓			Accessibility matrix		✓		✓	
[80], 2020	A1	PM4Py library based Prototype	✓			Sequence of events		✓			
[33], 2020	A5	ProM	✓	✓	✓	Sequence of events	✓	✓	✓	✓	✓
[81], 2020	A5	ProM	✓	✓		Sequence of events	✓				

3.3. C2. Tools

What tools are available for the event logs preprocessing task?

Tools for event log preprocessing are generally included as part of some tools for process mining. However, many commercial software tools for process mining (Perceptive Process Mining by Lexmark [82], Interstage Business Process Manager Analytics by Fujitsu Ltd. [83], Minit by Gradient ECM [84], myInvenio by Cognitive Technology [85], etc.) do not support event logs preprocessing tasks that help improving the quality of event logs. There are particular tools, applications, or frameworks developed for specific

preprocessing tasks of event logs [26,36,37,48,52,53,72]. Most of these tools are limited to a single process modeling language and use some type of data deployment or transformation. Moreover, there are specialized tools such as ProM [16], Apromore [86], Celonis [87], and RapidProm [88] that include different filters, routines, and algorithms for preprocessing the event log to support process mining tasks.

According to Will van der Aalst. [8], there are three categories of process mining tools that contain event log preprocessing. Type-1 process mining tools are mainly built for answering ad-hoc questions about event log preprocessing. An example of this tool type is Disco [89], which allows the user to interactively filter the data and project that data immediately on a newly learned process model. In Type-2 process mining tools, the analytic workflow is made explicit; that is, the user can visualize and decide what elements to isolate or eliminate from the event log. An example of this tool type is RapidProM. Finally, tools of Type-3 are tailored towards answering predefined questions repeatedly in a known setting. These tools are typically used to create "process dashboards" that provide standard views of process models. For example, the tool called Celonis Process Mining supports the creation of such process-centric dashboards.

Next, we describe some tools that include preprocessing or event log repair strategies as part of their functioning. Among the criteria considered to select these tools are their popularity in the process mining area (as they are reported in several papers) and the inclusion of preprocessing techniques.

The ProM framework [16] provides different event log filters (Filter event log based on choice, Filter events based on attribute value, filter log using simple heuristics, filter in high-frequency trace, among others) for cleaning event logs. These filters are especially useful when handling real-life logs and they do not only allow for projecting data in the log, but also for adding data to the log, removing process instances (cases), and removing and modifying events. There are several other filter plug-ins in ProM for the removal or repairing of activities, attributes, and events (Remove activities that never have utility, remove all attributes with value-empty, remove events without timestamps, refine labels globally, etc.). ProM is the most popular process mining tool that mostly has preprocessing techniques, since many of the research proposals are available from ProM. However, most of the available preprocessing techniques are focused on event filtering and trace clustering. ProM handles multiple formats and multiple languages, e.g., Petri nets, BPMN, EPCs, social networks, etc. Through the import of plug-ins, a wide variety of models can be loaded ranging from a Petri net to LTL formulas.

The ProM framework allows for interaction between a large number of plug-ins, i.e., implementations of algorithms and formal methods for analysis of business process, process mining, social network analysis, organizational mining, clustering, decision mining, prediction, and recommendation.

Apromore [86] is an open-source platform for advanced models of business processes. It allows applying a variety of filtering techniques to slice and dice an event log in different ways. There are two main filter types supported by Apromore: case filter and event filter. Both filter types allow creating a filter based on particular conditions on the cases or events. A case filter allows slicing a log, i.e., to retain a subset of the process cases. An event filter allows dicing a log, i.e., to retain a fragment of the process across multiple cases. There are other filters, such as timeframe that allows retaining or removing those cases that are active in, contained in, started in, or ended in a particular period of time. Another filtering technique is the rework repetition filter that can be used for process sequences containing certain repetitions are removed.

RapidProM [17] is an extension of RapidMiner, where the process mining framework ProM is integrated within RapidMiner to combine the best of both. In RapidProM, complex process mining workflows can be modeled, executed, and subsequently reused for other data sets. This tool includes data cleaning and filtering methods to filter cases based on their throughput time, with the possibility of choosing a different performance annotation. The RapidProM operators focus on the analysis of event data and process models. These

operators consider that events are related to process instances, and they should be handled as such.

Disco [89] is a process mining commercial tool that provides non-destructive filtering capabilities for explorative drill-down, and for focusing the analysis. These filters are accessible from any view and are easy to configure. They allow drilling down by case performance, time frame, variation, attributes, event relationships, or endpoints.

Celonis [87] uses machine learning to determine the specific root causes of deviations from a business process. This tool focuses on identifying inefficiencies or problems related with noisy events, or missing values through clustering and filtering algorithms.

Table 5 summarizes the main characteristics of the previous discussed tools. These tools are the most popular and widely known. They are tools that include preprocessing algorithms of the event logs studied in this survey, and allow process mining tasks (discovery, conformance, and enhancement) in combination with preprocessing algorithms within the same tool.

Table 5. Comparison of popular process mining tools incorporating event log preprocessing techniques.

Features	Prom 6.5.1	Apromore	RapidProM	Disco	Celonis
Trace/event Filtering	Yes	Yes	Yes	Yes	Yes
Trace clustering	Yes	Yes	Yes	No	Yes
Timestamp repair	Yes	Yes	Yes	Yes	No
Remove attributes, events or activities	Yes	Yes	Yes	Yes	No
Embedded preprocessing	Yes	No	Yes	No	No
Abstraction	Yes	No	No	No	No
Alignment	Yes	Yes	Yes	No	No

There are some other non commercial, automatic, or user-driven tools for repairing event logs. TimeCleanser [90] provides consolidated detection and repairing mechanisms to deal with data quality-related ordering issues in event logs. Li and Van der Aalst [91] present a framework for detecting deviations in complex event logs from control-flow perspective. This framework is based on an approach whose basic principle is that a case from a log is a deviation if it is not similar to the collection of mainstream cases in the log. In that work, the authors propose the creation of profiles rather than models or clusters, to detect deviations and improving the performance, iteratively. One can edit the profiling function to detect deviations from specific perspectives. However, in some situations, the approach does not handle properly the presence of loops in the models.

Interactive filtering [92] is an open source toolkit that enables the discovery of process models where users can filter iteratively infrequent behavior using different outlier filtering techniques (variant filtering, probabilistic methods, and sequential mining based method), and then to apply a discovery algorithm, such as the inductive visual miner and the interactive data-aware Heuristics miner. This tool shows the results of each change in thresholds or method on the discovered process model and allows user interaction.

Although there is an extensive list of commercial and free process mining tools that incorporate techniques for the preprocessing of event logs, so far, there is no tool that exclusively contains preprocessing strategies, capable of working with large event logs with different characteristics in a considerable time. Many of the tools that contain preprocessing techniques are limited to interacting with the user to make a better decision when including, isolating, or eliminating any event or trace.

3.4. C3. Representation Schemes of Event Logs Used in Preprocessing Techniques

What structures are more appropriate to represent and manipulate event logs in preprocessing techniques?

For years, the representation of information has been a basic need, almost in every domain, including process mining. Even though the total amount of storage space is not an important issue nowadays, since external memory (i.e., disk) can store huge amounts

of events, and is very cheap, the time required to access the event logs is an important bottleneck in many algorithms. An appropriate structure or representation scheme of the event logs will provide efficient management of large event logs supporting algorithms that process the events directly from the representation. One of the most common event log representations used in the preprocessing techniques is the vector space model (or bag-of-events) [43], where each trace is represented as a vector and each dimension corresponds to an event type. In this type of representation, the similarity between traces is measured using typical measures, such as Euclidean distance or Cosine similarity. Some proposed approaches for event log preprocessing use traces or event sequences as data structures for representation and manipulation of event logs, since they are simpler to filter, aggregate, or remove new events or traces on this structure. However, other structures, such as automatons, directed graphs, trace arrays, among others, have also been studied.

In [93], a graph repairing approach for detecting unsound structure, and repairing inconsistent event name is proposed. This approach repairs event data with inconsistent labeling but sound structure, using the minimum change principle to preserve the original information as much as possible. Then, an algorithm conducts the detection and repairing of dirty event data simultaneously, so that it either reports unsound structure or gives the minimum reparation of inconsistent event names. Moreover, an approximation algorithm, called PTIME, is presented in [93] to repair one transition at a time, which is repeatedly invoked until all violations are eliminated or no repairing can be further conducted.

Mueller-Wickop and Schultz [94] present an approach comprising four preprocessing steps for the reconstruction of process instance graphs to event log with a sequentially ordered list of activities by adding a directed sequence flow between activities of instance graphs. In this approach, instance graphs can be decomposed into independent parts, which can be mapped into a sequential event log. The first step is to mine the source data with the financial process mining (FPM) algorithm to obtain process instances represented as graphs. The second step consists of transforming these graphs to directed activity graphs. The third step is to enumerate all possible paths representing sub-sequences of the mined process instances. Finally, the fourth step is to store all paths in an event log representing the logical order of events.

According to the surveyed works, the most appropriate structure to represent, manipulate, and transform raw event logs is the traces or events sequence, which can be described as a vector of values whose implementation and use is simple. This structure also facilitates the tasks of inserting or deleting events or traces, mainly when working with cases or instances of a large length in the event log.

3.5. C4. Imperfection Types in the Event Log

What kind of imperfections are commonly identified in the event logs?

Chandra Bose et al. [3] establish that the most real-life event logs tend to be fine-granular, heterogeneous, voluminous, incomplete, and noisy, which can cause major problems in the different process mining tasks. On the one hand, fine granularity is related to the level of detail at which events are recorded. This can vary widely without considering the desired levels of analysis. This type of imperfection is closely related to the quality issue of data diversity (Section 2) where there are diverse types of records with different granularity levels due to a lack of good standards and guidelines for logging.

Often, the models produced from the discovery task are spaghetti-like and hard to comprehend due to fine granularity in the event log. Heterogeneity in the event log means that many of the real processes take place in diverse and unstructured environments, causing the generated event log to contain a heterogeneous mixture of these environments. Heterogeneity stems from operational processes that change over time to adapt to changing circumstances. The trace clustering techniques have been shown to be an effective way of dealing with heterogeneity.

Inside the fine granularity issue, there are also imperfections in the event logs related to coarse granular timestamps, which implies that the ordering of events within the log may not conform to the actual ordering in which the events occurred in reality; mixed granular timestamps, where there are events for which the level of granularity of their timestamps is different; incorrect timestamps, scenarios where the recorded timestamp of (some or all) events in the log does not correspond to the real time at which the event occurred.

Moreover, there are other common imperfections present in the event log associated to the data quality issue, such as missing attribute values or events missing anywhere within the trace, although they occurred in reality. There may also be problems in the event log associated with ambiguity between events, where multiple events have the same activity name. Another issue is the overlapping activity, where an instance of an activity is started and before it is completed, another instance of the same activity is started. Moreover, the presence of noisy data/outliers is common; that is, rare, exceptional, or anomalous execution behavior.

Authors in [29] identified a series of imperfections patterns commonly encountered in preprocessing raw source logs. These patterns are defined from the problems that are identified when transforming raw data source logs into an event log that is 'clean' and usable for process mining analysis. Some of the imperfection patterns presented in [29] are directly related to the issue of missing event values or noisy values, for example when the events in a log are not explicitly linked to their respective case identifiers, or when there are key process steps missing in the event log being analyzed but are recorded elsewhere. To address this type of imperfection pattern, most of the event- or trace-level filtering techniques studied in Section 3.2.1 attempt to identify missing events and correct them, or eliminate outlier events from the event log.

Another type of imperfection pattern also presented in [29] is related to problems in the timestamp attribute. This imperfection occurs when recording errors in the timestamp, or when the timestamp values are recorded in a different format from the expected (data diversity), or when recording events from electronic forms, such as the difference in the order of the events that were executed. Techniques to address the issues in the timestamp are mainly based on determining the impact that the timestamp information has to improve the quality of the event log.

In addition to the aforementioned patterns, there are also patterns related to problems in the labels associated with the events, such as the presence of a group of values (of certain attributes in an event log) that are syntactically different, but semantically similar, or the existence of two or more values of an event attribute that do not have an exact match with each other, but have strong similarities, syntactically and semantically. To address this type of imperfection pattern, the abstraction techniques and clustering turn out to be the most suitable for transforming event labels to a higher level of granularity, allowing to bridge the gap between an original low-level event log and a desired high-level perspective on the log.

Other authors [12] have identified that there are indicators associated with the time to detect imperfections in the order of the events of a log. Among the identified indicators are: (1) the existence of either coarse timestamp granularity or mixed timestamp value granularity from multiple systems, where each system records timestamps differently. An example of this is when an event x may be recorded at day-level granularity. Within the same case, another event y may have second-level granularity. The ordering of these two events will be incorrect; (2) identifying events exhibiting unusual temporal ordering (e.g., duplicate entry of exactly the same event; (3) learning the temporal position of a particular activity in the context of other activities, or the distribution of timestamp values of all events in a log may indicate the existence of timestamp-related problems. For example, when a log is comprised of events from multiple systems, there may be more than one way in which timestamps are formatted, which may lead to the 'misfielded' or 'unanchored' timestamp problem.

Despite the diversity of imperfections that may be present in the event log, and according to the review of the state-of-the-art, two of the most common problems are those related to the presence of noisy data, as well as the data diversity in the event log that deviates from the expected behavior.

3.6. C5. Related Tasks

What are the tasks closely related to event log preprocessing?

From the state-of-the-art works discussed so far, we identified two tasks strongly related to the data preprocessing in process mining: (1) event abstraction and (2) alignment. Both tasks allow improving the quality of the event log or the process model and the performance of some process mining techniques.

3.6.1. Event Abstraction

The majority of available process mining techniques assume that event data are captured on the same level of granularity. However, information systems in the real world record events at different granularity levels [95]. In many cases, events recorded in one event log are presented in a fine-grained level, causing process techniques and particularly process discovery algorithms to produce incomprehensible process models or models not representative of the event log. In these cases, the event abstraction techniques transform the event log to a higher level of granularity, allowing to bridge the gap between an original low-level event log and a desired high-level perspective on the log, such that more comprehensible process models can be discovered.

Some techniques proposed for event abstraction make use of supervised learning when annotations with high-level interpretations of the low-level events are available for a subset of the sequences (i.e., traces). These annotations provide guidance on how to label higher level events and guidance for the target level of abstraction. A general approach to supervised abstraction of events takes two inputs: (1) a set of annotated traces; that is, traces where the high-level event to which a low level event belongs (the label attribute of the low-level event) is known for all low-level events in the trace; and (2) a set of unannotated traces; that is, traces where the low level events are not mapped to high-level events.

Tax et al. [77] propose a method to abstract events in a XES event log that is too low-level, based on supervised learning and a condition random field learning step. A high-level interpretation of a low-level event log is achieved through a supervised learning model on the set of traces where high-level target labels are available, and applying the model to other low-level traces is possible to classify them. The recognition of high-level event labels is viewed as a sequence labeling task in which each event is classified as one of the higher-level events from a high-level event alphabet. That work proposes a sequence-focused metric to evaluate supervised event abstraction results that fits closely to the tasks of process discovery and conformance checking. Conditional random fields are trained from the annotated traces to create a probabilistic mapping from low-level events to high-level events. This mapping, once obtained, can be applied to the unannotated traces in order to estimate the corresponding high-level event for each low-level event.

Sun and Bauer [73] propose a process model abstraction technique to optimize the quality of the potential high level model and to consider the quality of the submodels generated where each sub-model is employed to show the details of its relevant high level activity in the high level model.

There are some others methods explored within the process mining field that address the challenge of abstracting low-level events to higher level events [64,65,69,73,74]. Existing event abstraction methods rely on unsupervised learning techniques [76,78] for clustering of low-level events into one high-level event. Current techniques require the user/process analyst to provide high-level event labels themselves based on domain knowledge, or generate long labels by concatenating the labels of all low-level events incorporated in the cluster. Many existing unsupervised event abstraction methods contain one or more parameters to control the degree in which events are clustered into higher level events.

Finding the right level of abstraction that provides meaningful results is often a matter of trial-and-error.

3.6.2. Alignment

Alignment requires that the events in the log are correlated with the activities in the process model, and the discrepancies and conformance degrees between a log and a model can be determined. Alignment is not only relevant for the conformance checking task, but also for event log repairing and model repairing. On the one hand, the event log repairing concentrates on repairing event sequences according to a predefined process model, while model repairing focuses on repairing the necessary parts of the process model, such that it can replay event sequences in the log [75].

A sequential alignment between a trace and a process model is defined as a sequence of moves, each relating an event in the trace to an activity in the model. A cost function assigns a cost to each possible move. A sequential alignment with the lowest cost, according to the cost function is an optimal alignment. Unfortunately, the problem of finding the optimal alignment is NP-hard [96]. This indicates that when the process is large, it is intractable to determine the optimal alignment. The optimal alignment occurs when a trace in the log and an occurrence sequence in the model have the shortest edit distance.

Rogge-Solti et al. [68] present a cost-based alignment approach to repair missing entries in the logs. The authors employ probabilistic filtering models to derive the most likely timestamp of missing events using path probabilities and stochastically enriched process models. Bayesian networks are used to capture the dependencies between the random durations by a stochastic Petri net.

Song et al. [71] propose an approach for recovering missing events in process logs by decomposition of process into different sub-processes and heuristics to prune unqualified sub-processes that fail to generate the minimum recovery. In order to reduce the redundant recoveries in regard to parallel routings, the authors use an algorithm that leverages trace replaying to efficiently find a minimum recovery.

De Leoni et al. [67] propose a conformance checking approach based on the principle of finding an alignment of a given event log and its corresponding process model. The A* algorithm is used to find, for each trace in the event log, an optimal alignment, i.e., an alignment that minimizes the cost of the deviations. The authors adapt alignment-based approaches to be able to deal with the large search spaces induced by the inherent flexibility of declarative models. Based on such alignments, they provide diagnostics, at the trace level, showing why events need to be inserted/removed in a trace, and at the model level, coloring constraints, and activities in the model based on their degree of conformance.

In [75], the authors use the structural and behavioral features of process models, effective heuristics based on process decomposition and trace replaying to reduce the search space for the optimal alignment. They use a divide-and-conquer strategy. The alignment requires that the events in the log are correlated with the activities in the process model. A general framework is developed to seek the optimal alignment between process models and event logs with missing, redundant, and dislocated events and activities, respectively. This framework is not only used to align event logs with process models, but is also utilized for repairing event logs. Their approach is realized in the tool Effa, which acts as a plugin of ProM. Some other works on alignment are presented in [63,66,70].

3.7. C6. Information Type

What type of attributes or information the preprocessing techniques use to work?

According to the grouping of event log preprocessing techniques presented in Section 3.2.1, many of these techniques use various information resources to derive improvements in the quality of event logs of the mined or already built models. On the one hand, the event log is considered the base element to be exploited for the process mining tasks, particularly the automatic discovery of process models, as well as the reference source in the conformance with the already built model. Many of the strategies for data preprocessing in process

mining are focused on working with the set of instances or traces that are obtained from the collection of events registered for executions of the process in question. To this end, different approaches that work on imperfections located in this source have been proposed. Some works have focused on some of the attributes identified in the collection of events, such as the information from timestamp which defines the ordering of events within the trace. Hence, incorrect ordering of events can have adverse effects on the outcome of process mining analysis. Even, in some scenarios in process mining, the timestamp attribute has been used to identify irregular processes using activity-level durations and contextual information. Some of the attributes used during the preprocessing are: case ID, event label, timestamp, cost, resource, contextual information, additional event payload, among others (see Figure 7). If the information of these attributes is missing, incomplete, in disorder, noisy or inconsistent, the preprocessing techniques clean and repair them.

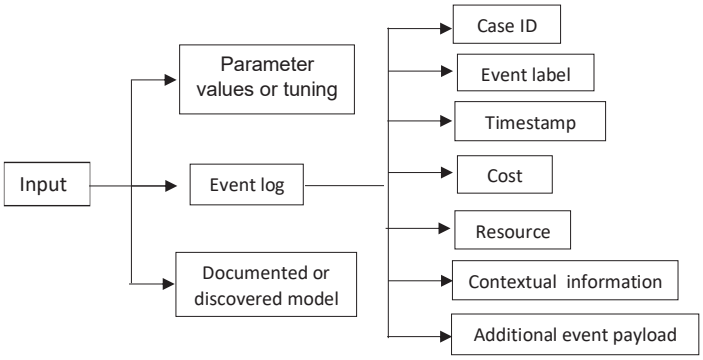

Figure 7. Input data used by most of the preprocessing techniques.

Tax et al. [15] use the identification of chaotic activities to improve the quality of the event log. These activities are described as activities that do not have a clear position in the process model and can occur spontaneously at any point in the process execution. If it is the case, the discovery of the rest of the process becomes complex. Some other works on event log preprocessing techniques [32,49,50,71] make use of the running process model that has been previously built by the organization's specialists.

Furthermore, most of the preprocessing techniques require certain parameter values or tuning for filtering or accepting events. This allows establishing certain decision thresholds and thus being able to determine if any event, trace or activity can be considered as inconsistent or noisy.

Many of the noise event detection and visualization techniques, particularly the clustering techniques (Section 3.2.2), use a traces set as input, which facilitates the division of the original event log. From this set, distances between traces are calculated, so that traces that have high similarity are grouped within the same cluster and instances that have low similarity are in different groups. Some clustering strategies exploit the information associated with events, such as resource and contextual information to improve the partitioning of the event log. In the case of pattern-based preprocessing techniques, they mainly use the raw event log to identify concrete forms, which keeps recurring non-arbitrary contexts, with the timestamp attribute being the most used by these techniques. Within the transformation techniques (filtering), it is common to use a set of traces to identify problems associated with the missing or noisy values contained in the different attributes in the event log.

Table 6 presents the relationships among the different characteristics (C1—techniques, C2—tools, C3—representation schemes, C4—imperfection types, C5—related tasks, and C6—types of information) of the preprocessing techniques surveyed in this work. As can be seen in the Table 6, filtering-based techniques are available in most of the process mining tools. However, the pattern-based techniques are only available through the ProM tool. Most of the processing techniques of the different classes handle the sequences of traces/events as their representation scheme of event logs to easily apply transformations

on the records. In this way, the traces are information resources that are mostly exploited in the preprocessing task. In addition, all preprocessing techniques consider the identification, isolation, and elimination of noise data, and to a lesser extent, the solution of problems related to missing, duplicate, and irrelevant data.

Table 6. Characteristics (C1–C6) on data preprocessing in the context of process mining.

Techniques (C1)	Tools (C2)	Representation Schemes (C3)	Imperfection Types (C4)	Related Tasks (C5)	Information Type (C6)
Filtering-based	ProM, Apromore, RapidProM, Disco, Celonis	sequences of traces/ activities	noise and missing data	alignment	traces
Time-based	ProM, Apromore, RapidProM, Disco	graph structure and sequences of events	missing, noise, diverse, and duplicate data	abstraction	time attribute
Clustering	ProM,RapidProM Disco, Celonis	sequences of traces/ events	noise and diversity data	abstraction	traces
pattern-based	ProM	raw event log	noise and diversity data	abstraction/ alignment	traces

4. Lessons Learned and Future Work

Based on the literature review, some important outcomes and guidelines can be inferred.

There is increasing interest in the study of preprocessing techniques for process mining from various domains (health, manufacturing, industry, etc.). They have demonstrated great success in building process models that are more simple to interpret and manipulate, causing many organizations to be interested in these types of techniques. This is more evident with the arrival of big data, having business processes with huge event logs, which could contain a high amount of imperfections and errors, such as missing values, duplicate events, evolutionary changes, fine-granular events, heterogeneity, noisy data outliers, and scoping. In this sense, the preprocessing techniques in process mining represent a fundamental basis to improve the execution and performance of process mining tasks required by experts in process models.

In practice, process mining requires more than one type of preprocessing technique to improve the quality of the event log (as shown in column 2 of Table 4). This is because an event log can have different data cleaning requirements and a single technique could not address all possible issues. For example, if the event log is voluminous; that is, with a large number of events or cases, a suitable technique for this type of log is trace-clustering. This preprocessing technique divides the original log into small sub-logs, allowing to reduce the complexity of its handling and storage. If the event log size is of average size (normal), but there is high variability in the size of the set of traces that are formed from the log, it is highly possible that filtering techniques at the event/trace level are more suitable. On the other hand, in those event logs, where it is estimated that the duration of the activities of an event is too slow or too fast, the use of preprocessing techniques based on the study of the timestamp is suggested.

From the review presented in this work, it is observed that the most commonly used preprocessing techniques are trace-clustering, and trace/event level filtering (see Figure 8), mainly due to the fact that they are easy to implement and adequately manage noise and incompleteness in the event logs, and also allow models to be identified from less-structured processes. On the one hand, the trace clustering technique is more suitable for the case where it is required to reduce the complexity of the discovered models. This technique is generally applied together with pattern identification or event abstraction techniques, since both are strongly linked to identifying associations or rules from observed behaviors, or acquired experiences in the event log. On the other hand, trace/event filtering techniques are sometimes applied in conjunction with timestamp-based techniques to achieve the identification and correction of missing or noisy values in the event log.

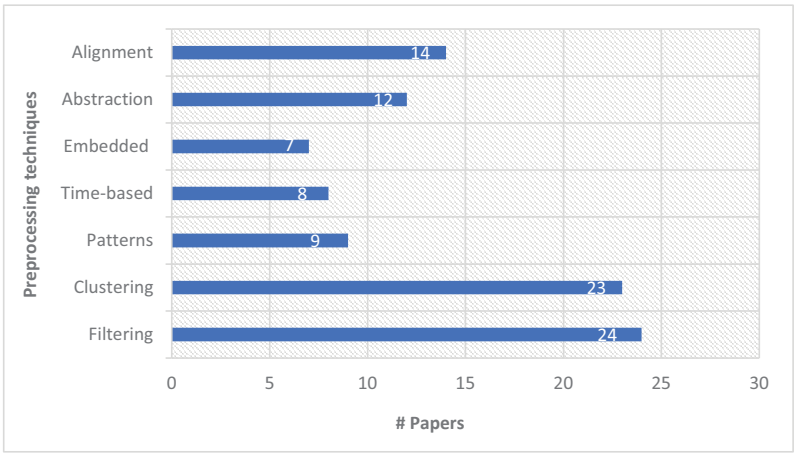

Figure 8. Preprocessing techniques and their distribution according to the proposed classification in this work.

Several works on data preprocessing in process mining focus on the identification of specific noise patterns associated with the quality of the event log. For example, in the method proposed by Hsu et al. [30], 21 irregular process instances from a set of 2169 were identified. The results were presented to a group of domain knowledge experts who confirmed that 81% of the identified process instances were abnormal. By contrast, only 9% of the identified outlier process instances by the proposed method were confirmed as outliers in the same environment setting. This and other works have considered event logs available in the literature or with common characteristics. However, the study of several event logs in different scenarios considering different characteristics (log size, number of attributes, resources, organizations, among others) could be considered for the identification of new noise patterns that have not been previously identified in the studied event logs.

Today, there are no popular or widely known preprocessing tools fully dedicated to solving the preprocessing tasks that allow working with repositories and event logs of different characteristics, independently of the process mining task that will use that preprocessing. Therefore, the design and implementation of new tools dedicated to data preprocessing for process mining is required. These tools could incorporate a kind of "intelligence" and interact with the user to decide which events to correct or not. ProM is the most common tool in process mining used to incorporate new plugins of preprocessing techniques.

According to the surveyed works, it has been possible to identify a greater need for the application of preprocessing techniques in the process mining tasks, mainly in the discovery of process models [56,57,59–62] and the conformance verification. On the one hand, in the discovery of process models, the preprocessing can reduce the complexity of the mined models through the identification, correction, and elimination of errors associated with event logs for the correct identification of the model gateways and, therefore, allows the discovery of more structured models. This would facilitate the interpretation of the discovered models, trying to maintain the original behavior of the event log. On the other hand, the preprocessing techniques have used for the conformance verification task between the event log and the discovered model. This is required to make a correct mapping between a clean event log and free of events, activities or traces that are missing, noisy, or inconsistent with the model in execution. In addition, the conformance task between the event log and the model can be executed in a considerable time, especially when there are large event logs, always expecting to get an output result, in the case where an enhancement task is focused on extending or improving an existing process model, using information from the actual model recorded in an event log, including, to a lesser degree, the use of preprocessing techniques.

Some surveyed works report measures related to the lack of quality in the event logs, such as number of missing traces, the ratio of identified irregularities, and presence or absence of imperfection patterns. However, the vast majority of works report measures related to the quality of the discovered models (fitness, recall, precision, and f-measure) with the raw even log and preprocessed event log. Few works report any study or result of the computational complexity of their proposals. These works mainly report the execution time of their algorithms, which can be highly variable depending on the different variables used in the calculation (size of the log, search algorithm, size of the traces, types of attributes of the log, etc.).

5. Conclusions

In this survey, we presented, for the first time, a literature review about the main approaches used in data preprocessing for process mining. The review included a description of techniques and algorithms, tools, frequently posed questions, perspectives, and data types. Representative works were systematically revised to determine the key aspects in the preprocessing techniques that lead to improve the quality of a process model. As a result, this paper provided, for the first time, a grouping of the different existing preprocessing techniques. This grouping is organized in transformation techniques and detection-visualization techniques. Transformation techniques carry out actions to mark changes in the original structure of the raw event log in order to improve the quality of the log. While the detection–visualization techniques identify, group, and isolate those events or traces that can generate problems in the quality of the event log.

We also presented the challenges that must be addressed by these techniques. Furthermore, this survey presents some of the key elements to consider for data preprocessing in process mining: (1) grouping of existing techniques for the preprocessing of event logs; (2) preprocessing tools in the context of process mining available in the literature; (3) the more appropriate data structures to represent and manipulate event logs in preprocessing techniques; (4) the problems and imperfections more often found in event logs; (5) the tasks more often related to event log preprocessing; and (6) the type of attributes or information that the preprocessing techniques use to work.

This review can serve as a reference guide to identify the different types of preprocessing techniques and their qualities. Moreover, it seeks to highlight the foundations for those aspects that should be taken into account to obtain process models being simple and easy to interpret.

Concluding this work, we can affirm that data preprocessing in the context of process mining will continue to be a topic of great interest. In the following years, with the arrival of big data and the internet of things, in the creation of huge event logs, it may be necessary to design new preprocessing algorithms that deal with new challenges that have not, so far, been identified or solved. Great progress has been made with clustering and data filtering techniques in process mining; however, other techniques, such as those based on the identification of imperfection patterns, have not yet fully addressed the automatic identification of imperfection patterns.

As part of future work to consider, a suite of metrics describing the existence of noise patterns in event logs should be developed. The noise patterns could affect an event log at different levels, including attributes, events, and cases; thus, the various pervasiveness metrics would determine the number of attributes, events, and cases affected by individual noise patterns and the noise pattern collection overall. Few works [29,33] have tried to identify and quantify the level of imperfection that exists in an event log. However, until now, there are no specific metrics that determine this level. Another topic of possible interest, as part of the continuation of this line of research, consists of having frameworks that not only compute the accuracy or fitness obtained by the model to evaluate the impact of the preprocessing being used, but also the computational costs, memory, and time complexity associated with data cleaning.

Author Contributions: Conceptualization, H.M.M.-C.; Formal analysis, H.M.M.-C.; Investigation, H.M.M.-C.; Methodology, H.M.M.-C.; Project administration, H.M.M.-C. and E.T.-L.; Supervision, H.M.M.-C.; Writing-original draft preparation, H.M.M.-C.; Writing-review and editing, H.M.M.-C. and E.T.-L.; Funding acquisition, E.T.-L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Autonomous University of Tamaulipas, Mexico. This research was also supported by Mexico's National Council of Science and Technology (CONACYT) under the Cátedras CONACYT project number 214/2016.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dakic, D.; Stefanović, D.; Cosic, I.; Lolić, T.; Medojevic, M. Business Process Mining Application: A Literature Review. In Proceedings of the 29th International DAAAM Symposium 2018, Zadar, Croatia, 24–27 October 2018; pp. 0866–0875. [CrossRef]
2. van der Aalst, W. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 136. [CrossRef]
3. Bose, R.P.J.C.; Mans, R.S.; van der Aalst, W.M.P. Wanna improve process mining results? In Proceedings of the 2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Singapore, 16–19 April 2013; pp. 127–134. [CrossRef]
4. Mans, R.S.; van der Aalst, W.M.P.; Vanwersch, R.J.B.; Moleman, A.J. Process Mining in Healthcare: Data Challenges When Answering Frequently Posed Questions. In *Process Support and Knowledge Representation in Health Care*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 140–153.
5. Emamjome, F.; Andrews, R.; ter Hofstede, A.; Reijers, H. Alohomora: Unlocking data quality causes through event log context. In Proceedings of the 28th European Conference on Information Systems (ECIS2020), Marrakech, Morocco, 15–17 June 2020; Association for Information Systems: Atlanta, GA, USA, 2020; pp. 1–16.
6. Batini, C.; Scannapieco, M. *Data Quality: Concepts, Methodologies and Techniques*; Springer: Berlin/Heidelberg, Germany, 2006. [CrossRef]
7. Wand, Y.; Wang, R.Y. Anchoring Data Quality Dimensions in Ontological Foundations. *Commun. ACM* **1996**, *39*, 86–95. [CrossRef]
8. van der Aalst, W.M.P. *Process Mining-Data Science in Action*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2016. [CrossRef]
9. Wang, J.; Song, S.; Zhu, X.; Lin, X.; Sun, J. Efficient Recovery of Missing Events. *IEEE Trans. Knowl. Data Eng.* **2016**, *28*, 2943–2957. [CrossRef]
10. Conforti, R.; Rosa, M.L.; ter Hofstede, A.H.M. Filtering Out Infrequent Behavior from Business Process Event Logs. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 300–314. [CrossRef]
11. van Zelst, S.J.; Fani Sani, M.; Ostovar, A.; Conforti, R.; La Rosa, M. Filtering Spurious Events from Event Streams of Business Processes. In *Advanced Information Systems Engineering*; Krogstie, J., Reijers, H.A., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 35–52.
12. Dixit, P.M.; Suriadi, S.; Andrews, R.; Wynn, M.T.; ter Hofstede, A.H.M.; Buijs, J.C.A.M.; van der Aalst, W.M.P. Detection and Interactive Repair of Event Ordering Imperfection in Process Logs. In *Advanced Information Systems Engineering*; Krogstie, J., Reijers, H.A., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 274–290.
13. Fani Sani, M.; van Zelst, S.J.; van der Aalst, W.M.P. Repairing Outlier Behaviour in Event Logs. In *Business Information Systems*; Springer International Publishing: Cham, Switzerland, 2018; pp. 115–131.
14. Sani, M.F.; van Zelst, S.J.; van der Aalst, W.M.P. Applying Sequence Mining for Outlier Detection in Process Mining. On the Move to Meaningful Internet Systems. In Proceedings of the OTM 2018 Conferences-Confederated International Conferences: CoopIS, C&TC, and ODBASE 2018, Valletta, Malta, 22–26 October 2018; pp. 98–116.
15. Tax, N.; Sidorova, N.; van der Aalst, W.M.P. Discovering more precise process models from event logs by filtering out chaotic activities. *J. Intell. Inf. Syst.* **2019**, *52*, 107–139. [CrossRef]
16. Van Dongen, B.F.; de Medeiros, A.K.A.; Verbeek, H.; Weijters, A.; Van Der Aalst, W.M. The ProM framework: A new era in process mining tool support. In *International Conference on Application and Theory of Petri Nets*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 444–454.
17. van der Aalst, W.M.P.; Bolt, A.; van Zelst, S.J. RapidProM: Mine Your Processes and Not Just Your Data. *arXiv* **2017**, arXiv:1703.03740.
18. Bezerra, F.; Wainer, J. Anomaly Detection Algorithms in Logs of Process Aware Systems. In Proceedings of the 2008 ACM Symposium on Applied Computing, SAC'08, Fortaleza, Brazil, 16–20 March 2008; ACM: New York, NY, USA, 2008; pp. 951–952. [CrossRef]

19. Jalali, H.; Baraani, A. Genetic-based anomaly detection in logs of process aware systems. *World Acad. Sci. Eng. Technol.* **2010**, *64*, 304–309.
20. de Lima Bezerra, F.; Wainer, J. A Dynamic Threshold Algorithm for Anomaly Detection in Logs of Process Aware Systems. *JIDM* **2012**, *3*, 316–331. [CrossRef]
21. Bezerra, F.; Wainer, J. Algorithms for Anomaly Detection of Traces in Logs of Process Aware Information Systems. *Inf. Syst.* **2013**, *38*, 33–44. [CrossRef]
22. Cheng, H.; Kumar, A. Process mining on noisy logs-Can log sanitization help to improve performance? *Decis. Support Syst.* **2015**, *79*, 138–149. [CrossRef]
23. Sani, M.F.; van Zelst, S.J.; van der Aalst, W.M.P. Improving Process Discovery Results by Filtering Outliers Using Conditional Behavioural Probabilities. In Proceedings of the Business Process Management Workshops-BPM 2017 International Workshops, Barcelona, Spain, 10–11 September 2017; pp. 216–229. [CrossRef]
24. Bezerra, F.; Wainer, J.; van der Aalst, W.M.P. Anomaly Detection Using Process Mining. In *Enterprise, Business-Process and Information Systems Modeling*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 149–161.
25. Böhmer, K.; Rinderle-Ma, S. Multi-perspective Anomaly Detection in Business Process Execution Events. In Proceedings of the On the Move to Meaningful Internet Systems: OTM 2016 Conferences, Rhodes, Greece, 24–28 October 2016; Springer International Publishing: Cham, Switzerland, 2016; pp. 80–98.
26. Kong, L.; Li, C.; Ge, J.; Li, Z.; Zhang, F.; Luo, B. An Efficient Heuristic Method for Repairing Event Logs Independent of Process Models. In Proceedings of the 4th International Conference on Internet of Things, Big Data and Security, Heraklion, Crete, Greece, 2–4 May 2019; Volume 1: IoTBDS, INSTICC; SciTePress: Setúbal, Portugal, 2019; pp. 83–93. [CrossRef]
27. Sani, M.F.; van Zelst, S.J.; van der Aalst, W.M.P. Repairing Outlier Behaviour in Event Logs using Contextual Behaviour. *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.* **2019**, *14*, 5:1–5:24. [CrossRef]
28. Song, S.; Cao, Y.; Wang, J. Cleaning Timestamps with Temporal Constraints. *Proc. VLDB Endow.* **2016**, *9*, 708–719. [CrossRef]
29. Suriadi, S.; Andrews, R.; ter Hofstede, A.H.; Wynn, M.T. Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. *Inf. Syst.* **2017**, *64*, 132–150. [CrossRef]
30. Hsu, P.Y.; Chuang, Y.C.; Lo, Y.C.; He, S.C. Using contextualized activity-level duration to discover irregular process instances in business operations. *Inf. Sci.* **2017**, *391–392*, 80–98. [CrossRef]
31. Tax, N.; Alasgarov, E.; Sidorova, N.; van der Aalst, W.M.P.; Haakma, R. Time-Based Label Refinements to Discover More Precise Process Models. *arXiv* **2017**, arXiv:1705.09359. [CrossRef]
32. Andreas, R.S.; Ronny, S.; van der Aalst, W.M.P.; Mathias, W. Repairing Event Logs Using Timed Process Models. In Proceedings of the On the Move to Meaningful Internet Systems: OTM 2013 Workshops, Graz, Austria, 9–13 September 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 705–708.
33. Fischer, D.A.; Goel, K.; Andrews, R.; van Dun, C.; Wynn, M.; Röglinger, M. Enhancing Event Log Quality: Detecting and Quantifying Timestamp Imperfections. In Proceedings of the Business Process Management: 18th International Conference BPM 2020, Seville, Spain, 13–18 September 2020; Springer: Cham, Switzerland, 2020; Volume 392, pp. 309–326.
34. Song, M.; Günther, C.W.; van der Aalst, W.M.P. Trace Clustering in Process Mining. In *Business Process Management Workshops*; Ardagna, D., Mecella, M., Yang, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 109–120.
35. Bose, R.P.J.C.; van der Aalst, W.M.P. Trace Clustering Based on Conserved Patterns: Towards Achieving Better Process Models. In *Business Process Management Workshops*; Rinderle-Ma, S., Sadiq, S., Leymann, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 170–181.
36. Evermann, J.; Thaler, T.; Fettke, P. Clustering Traces Using Sequence Alignment. In *Business Process Management Workshops*; Reichert, M., Reijers, H.A., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 179–190.
37. Xu, J.; Liu, J. A Profile Clustering Based Event Logs Repairing Approach for Process Mining. *IEEE Access* **2019**, *7*, 17872–17881. [CrossRef]
38. Bose, R.P.J.C.; van der Aalst, W.M.P. Context Aware Trace Clustering: Towards Improving Process Mining Results. In Proceedings of the SIAM International Conference on Data Mining, SDM 2009, Sparks, NV, USA, 30 April–2 May 2009; pp. 401–412. [CrossRef]
39. Wang, X.; Zhang, L.; Cai, H. Using Suffix-Tree to Identify Patterns and Cluster Traces from Event Log. In *Signal Processing and Information Technology*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 126–131.
40. Jagadeesh Chandra Bose, R. Process Mining in the Large: Preprocessing, Discovery, and Diagnostics. Ph.D. Thesis, Department of Mathematics and Computer Science, Eindhoven, The Netherlands, 2012. [CrossRef]
41. Hompes, B.; Buijs, J.; van der Aalst, W.; Dixit, P.; Buurman, J. Discovering deviating cases and process variants using trace clustering. In Proceedings of the 27th Benelux Conference on Artificial Intelligence, Hasselt, Belgium, 5–6 November 2015.
42. Sun, Y.; Bauer, B.; Weidlich, M. Compound Trace Clustering to Generate Accurate and Simple Sub-Process Models. In *Service-Oriented Computing*; Springer International Publishing: Cham, Switzerland, 2017; pp. 175–190.
43. Greco, G.; Guzzo, A.; Pontieri, L.; Sacca, D. Discovering expressive process models by clustering log traces. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 1010–1027. [CrossRef]
44. Ferreira, D.; Zacarias, M.; Malheiros, M.; Ferreira, P. Approaching Process Mining with Sequence Clustering: Experiments and Findings. In Proceedings of the 5th International Conference on Business Process Management, BPM'07, Brisbane, Australia, 24–28 September 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 360–374.

45. de Medeiros, A.K.A.; Guzzo, A.; Greco, G.; van der Aalst, W.M.P.; Weijters, A.J.M.M.; van Dongen, B.F.; Saccà, D. Process Mining Based on Clustering: A Quest for Precision. In *Business Process Management Workshops*; ter Hofstede, A., Benatallah, B., Paik, H.Y., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 17–29.
46. De Weerdt, J.; vanden Broucke, S.; Vanthienen, J.; Baesens, B. Active Trace Clustering for Improved Process Discovery. *IEEE Trans. Knowl. Data Eng.* **2013**, *25*, 2708–2720. [CrossRef]
47. Nguyen, P.; Slominski, A.; Muthusamy, V.; Ishakian, V.; Nahrstedt, K. Process Trace Clustering: A Heterogeneous Information Network Approach. In Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, FL, USA, 5–7 May 2016; pp. 279–287. [CrossRef]
48. Folino, F.; Greco, G.; Guzzo, A.; Pontieri, L. Mining Usage Scenarios in Business Processes: Outlier-aware Discovery and Run-time Prediction. *Data Knowl. Eng.* **2011**, *70*, 1005–1029. [CrossRef]
49. Chatain, T.; Carmona, J.; van Dongen, B. Alignment-Based Trace Clustering. In *Conceptual Modeling*; Mayr, H.C., Guizzardi, G., Ma, H., Pastor, O., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 295–308.
50. Boltenhagen, M.; Chatain, T.; Carmona, J. Generalized Alignment-Based Trace Clustering of Process Behavior. In *Application and Theory of Petri Nets and Concurrency*; Donatelli, S., Haar, S., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 237–257.
51. Ukkonen, E. On-Line Construction of Suffix Trees. *Algorithmica* **1995**, *14*, 249–260. [CrossRef]
52. Ghionna, L.; Greco, G.; Guzzo, A.; Pontieri, L. Outlier Detection Techniques for Process Mining Applications. In *Foundations of Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 150–159.
53. Chapela-Campa, D.; Mucientes, M.; Lama, M. Discovering Infrequent Behavioral Patterns in Process Models. In *Business Process Management*; Carmona, J., Engels, G., Kumar, A., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 324–340.
54. Jagadeesh Chandra Bose, R.P.; van der Aalst, W.M.P. Abstractions in Process Mining: A Taxonomy of Patterns. In *Business Process Management*; Dayal, U., Eder, J., Koehler, J., Reijers, H.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 159–175.
55. Günther, C.W.; van der Aalst, W.M.P. Fuzzy Mining–Adaptive Process Simplification Based on Multi-perspective Metrics. In *Business Process Management*; Alonso, G., Dadam, P., Rosemann, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 328–343.
56. Gu, C.-Q.; Chang, H.-Y.; Yi, Y. Workflow mining: Extending the alpha algorithm to mine duplicate tasks. In Proceedings of the 2008 International Conference on Machine Learning and Cybernetics, Kunming, China, 10 January 2008; Volume 1, pp. 361–368. [CrossRef]
57. Folino, F.; Greco, G.; Guzzo, A.; Pontieri, L. Discovering Expressive Process Models from Noised Log Data. In Proceedings of the 2009 International Database Engineering & Applications Symposium, IDEAS'09, Calabria, Italy, 16–18 September 2009; ACM: New York, NY, USA, 2009; pp. 162–172. [CrossRef]
58. Weijters, A.J.M.M.; Ribeiro, J.T.S. Flexible Heuristics Miner (FHM). In Proceedings of the 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Paris, France, 11–15 April 2011; pp. 310–317. [CrossRef]
59. Leemans, S.J.J.; Fahland, D.; van der Aalst, W.M.P. Discovering Block-Structured Process Models from Event Logs Containing Infrequent Behaviour. In Proceedings of the Business Process Management Workshops-BPM 2013 International Workshops, Beijing, China, 26 August 2013; pp. 66–78. [CrossRef]
60. Leemans, S.J.J.; Fahland, D.; van der Aalst, W.M.P. Discovering Block-Structured Process Models from Incomplete Event Logs. In *Application and Theory of Petri Nets and Concurrency*; Ciardo, G., Kindler, E., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 91–110.
61. Mannhardt, F.; de Leoni, M.; Reijers, H.A.; van der Aalst, W.M.P. Data-Driven Process Discovery—Revealing Conditional Infrequent Behavior from Event Logs. In *Advanced Information Systems Engineering*; Dubois, E., Pohl, K., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 545–560.
62. vanden Broucke, S.K.; Weerdt, J.D. Fodina: A robust and flexible heuristic process discovery technique. *Decis. Support Syst.* **2017**, *100*, 109–118. [CrossRef]
63. Jagadeesh Chandra Bose, R.P.; van der Aalst, W. Trace Alignment in Process Mining: Opportunities for Process Diagnostics. In *Business Process Management*; Hull, R., Mendling, J., Tai, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 227–242.
64. van Dongen, B.F.; Adriansyah, A. Process Mining: Fuzzy Clustering and Performance Visualization. In *Business Process Management Workshops*; Rinderle-Ma, S., Sadiq, S., Leymann, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 158–169.
65. Günther, C.W.; Rozinat, A.; van der Aalst, W.M.P. Activity Mining by Global Trace Segmentation. In *Business Process Management Workshops*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 128–139.
66. Bose, R.J.C.; van der Aalst, W.M. Process diagnostics using trace alignment: Opportunities, issues, and challenges. *Inf. Syst.* **2012**, *37*, 117–141. [CrossRef]
67. de Leoni, M.; Maggi, F.M.; van der Aalst, W.M.P. Aligning Event Logs and Declarative Process Models for Conformance Checking. In *Business Process Management*; Barros, A., Gal, A., Kindler, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 82–97.
68. Rogge-Solti, A.; Mans, R.S.; van der Aalst, W.M.P.; Weske, M. Improving Documentation by Repairing Event Logs. In *The Practice of Enterprise Modeling*; Grabis, J., Kirikova, M., Zdravkovic, J., Stirna, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 129–144.

69. Baier, T.; Mendling, J. Bridging Abstraction Layers in Process Mining by Automated Matching of Events and Activities. In *Business Process Management*; Daniel, F., Wang, J., Weber, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 17–32.
70. Lu, X.; Fahland, D.; van der Aalst, W.M.P. Conformance Checking Based on Partially Ordered Event Data. In *Business Process Management Workshops*; Fournier, F., Mendling, J., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 75–88.
71. Song, W.; Xia, X.; Jacobsen, H.; Zhang, P.; Hu, H. Heuristic Recovery of Missing Events in Process Logs. In Proceedings of the 2015 IEEE International Conference on Web Services, New York, NY, USA, 27 June–2 July 2015; pp. 105–112. [CrossRef]
72. Lu, X.; Fahland, D.; van der Aalst, W.M.P. Interactively Exploring Logs and Mining Models with Clustering, Filtering, and Relabeling. In Proceedings of the BPM Demo Track 2016 Co-located with the 14th International Conference on Business Process Management (BPM 2016), Rio de Janeiro, Brazil, 21 September 2016; pp. 44–49.
73. Sun, Y.; Bauer, B. A Graph and Trace Clustering-based Approach for Abstracting Mined Business Process Models. In Proceedings of the 18th International Conference on Enterprise Information Systems, ICEIS 2016, Rome, Italy, 25–28 April 2016; SCITEPRESS-Science and Technology Publications: Lda, Portugal, 2016; pp. 63–74. [CrossRef]
74. Mannhardt, F.; de Leoni, M.; Reijers, H.A.; van der Aalst, W.M.P.; Toussaint, P.J. From Low-Level Events to Activities—A Pattern-Based Approach. In *Business Process Management*; La Rosa, M., Loos, P., Pastor, O., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 125–141.
75. Song, W.; Xia, X.; Jacobsen, H.a.; Zhang, P.; Hu, H. Efficient Alignment Between Event Logs and Process Models. *IEEE Trans. Serv. Comput.* **2016**, *10*, 136–149. [CrossRef]
76. Mannhardt, F.; Tax, N. Unsupervised Event Abstraction using Pattern Abstraction and Local Process Models. In Proceedings of the Radar tracks at the 18th International Working Conference on Business Process Modeling, Development and Support (BPMDS), Essen, Germany, 12–13 June 2017; pp. 55–63.
77. Tax, N.; Sidorova, N.; Haakma, R.; van der Aalst, W.M.P. Event Abstraction for Process Mining Using Supervised Learning Techniques. In Proceedings of the SAI Intelligent Systems Conference (IntelliSys), London, UK, 21–22 September 2016; Bi, Y., Kapoor, S., Bhatia, R., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 251–269.
78. Alharbi, A.M. Unsupervised Abstraction for Reducing the Complexity of Healthcare Process Models. Ph.D. Thesis, School of Computing, University of Leeds, Leeds, UK, 2019.
79. Huang, Y.; Zhong, L.; Chen, Y. Filtering Infrequent Behavior in Business Process Discovery by Using the Minimum Expectation. *Int. J. Cogn. Informatics Nat. Intell. (IJCINI)* **2020**, *14*, 1–15. [CrossRef]
80. Vidgof, M.; Djurica, D.; Bala, S.; Mendling, J. *Cherry-Picking from Spaghetti: Multi-Range Filtering of Event Logs*; Lecture Notes in Business Information Processing; Springer: Cham, Switzerland, 2020; Volume 387, pp. 135–149. [CrossRef]
81. Denisov, V.; Fahland, D.; Aalst, W. Repairing Event Logs with Missing Events to Support Performance Analysis of Systems with Shared Resources. In *Petri Nets 2020*; Springer: Paris, France, 2020.
82. Workflow and Case Management, 2009. Available online: www.lexmark.com (accessed on 21 January 2021).
83. Interstage Business Process Manager Analytics By Fujitsu Ltd., 2009. Available online: www.fujitsu.com (accessed on 18 October 2020).
84. Minit By Gradient ECM, 2015. Available online: https://golden.com/wiki/Minit-5NNVAR (accessed on 20 April 2021).
85. myInvenio By Cognitive Technology, 2016. Available online: www.my-invenio.com (accessed on 21 January 2021).
86. T.A. Foundation. Apromore-Advanced Process Analytics Platform. The University of Melbourne, 2011. Available online: https://apromore.org/(accessed on 19 April 2021).
87. Celonis, S.E.; Munich, G. *Celonis Process Mining*; CELONIS: New York, NY, USA, 2009. Available online: https://www.celonis.com/ (accessed on 19 April 2021).
88. Mans, R.; van der Aalst, W.; Verbeek, H. Supporting process mining workflows with RapidProM. In Proceedings of the BPM Demo Sessions 2014 co-located with BPM 2014, Eindhoven, The Netherlands, 20 September 2014; Limonad, L., Weber, B., Eds.; pp. 56–60.
89. BV, F. Discover Your Processes. Fluxicon Process Mining for Professionals. 2011. Available online: https://fluxicon.com/disco/ (accessed on 20 April 2021).
90. Gschwandtner, T.; Aigner, W.; Miksch, S.; Gärtner, J.; Kriglstein, S.; Pohl, M.; Suchy, N. TimeCleanser: A visual analytics approach for data cleansing of time-oriented data. In Proceedings of the 14th International Conference on Knowledge Management and Data-Driven Business, I-KNOW'14, Graz, Austria, 16–19 September 2014; pp. 1–8. [CrossRef]
91. Li, G.; van der Aalst, W. A framework for detecting deviations in complex event logs. *Intell. Data Anal.* **2017**, *21*, 759–779. [CrossRef]
92. Sani, M.F.; Berti, A.; van Zelst, S.J.; van der Aalst, W.M.P. Filtering Toolkit: Interactively Filter Event Logs to Improve the Quality of Discovered Models. In Proceedings of the Dissertation Award, Doctoral Consortium, and Demonstration Track at BPM 2019, Vienna, Austria, 1–6 September 2019; Volume 2420, pp. 134–138.
93. Wang, J.; Song, S.; Lin, X.; Zhu, X.; Pei, J. Cleaning structured event logs: A graph repair approach. In Proceedings of the 2015 IEEE 31st International Conference on Data Engineering, Seoul, Korea, 13–17 April 2015; pp. 30–41. [CrossRef]
94. Mueller-Wickop, N.; Schultz, M. ERP Event Log Preprocessing: Timestamps vs. Accounting Logic. In *Design Science at the Intersection of Physical and Virtual Design*; vom Brocke, J., Hekkala, R., Ram, S., Rossi, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 105–119.

95. van Zelst, S.; Mannhardt, F.; de Leoni, M.; Koschmider, A. Event Abstraction in Process Mining -Literature Review and Taxonomy. *Granul. Comput.* **2020**. [CrossRef]
96. de Leoni, M.; van der Aalst, W.M.P. Aligning Event Logs and Process Models for Multi-perspective Conformance Checking: An Approach Based on Integer Linear Programming. In *Business Process Management*; Daniel, F., Wang, J., Weber, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 113–129.