2021

Process Mining to Discover and Preserve Infrequent Relations in Event Logs: An Application to Understand the Laboratory Test Ordering Process Using the MIMIC-III Dataset

Qifan Chen
The University of Sydney, qifan.chen@sydney.edu.au

Yang Lu
The University of Sydney, yang.lu@sydney.edu.au

Charmaine Tam
The University of Sydney, charmaine.tam@sydney.edu.au

Simon Poon
The University of Sydney, simon.poon@sydney.edu.au

Process Mining to Discover and Preserve Infrequent Relations in Event Logs: An Application to Understand the Laboratory Test Ordering Process Using the MIMIC-III Dataset

Full research paper

Qifan Chen
School of Computer Science
The University of Sydney
Sydney, Australia
Email: qifan.chen@sydney.edu.au

Yang Lu
School of Computer Science
The University of Sydney
Sydney, Australia
Email: yang.lu@sydney.edu.au

Charmaine S. Tam
Centre for Translational Data Science and Northern Clinical School
The University of Sydney
Sydney, Australia
Email: charmaine.tam@sydney.edu.au

Simon K. Poon
School of Computer Science
The University of Sydney
Sydney, Australia
Email: simon.poon@sydney.edu.au

Abstract

Process mining techniques can provide insights into the healthcare domain with the rapid growth of electrical health records. Process mining is about understanding the sequence of activities in event logs, where directly-follows relations identify pairs of activities that follow each other directly. Existing research explores frequent relations, while infrequent relations are often seen as noises and filtered out during discovery. However, important insights may be revealed through these infrequent relations, especially in healthcare processes. This paper aims to use process mining techniques to discover and preserve value-based conditional infrequent relations. We adopt the L* life-cycle methodology and Data-aware Heuristic Miner (DHM) as tools to provide a worded example based on extracted data from the MIMIC-III dataset, which is a publicly available database containing a large amount of electrical health records (EHR), to show how process mining can be used to analyse infrequent relations in a laboratory test's ordering process.

Keywords Process Mining, Healthcare Process, Conditional Infrequent Relation, MIMIC-III

1 Introduction

Process mining is a technology useful for understanding business processes by utilising event logs captured in information systems (Van Der Aalst 2016). The input for process discovery is the event log, which contains a collection of events, each associated with a timestamp and resources, that records necessary information such as the person responsible and the output when performing the event. This is exemplified in Figure 1. An event represents a unique execution of an activity, which is a well-defined task in a process, such as the different blood tests ordered in hospital (Mans et al. 2015). Cases group events, also called process instances or traces. Figure 1 illustrates a trace containing four events for a patient. Relations are specified as directly-follows relations in this paper, where they identify pair of activities that follow each other directly. Sodium⟶Glucose is an example of a relation in Figure 1. These relations are identified as frequent or infrequent based on their occurrence times in event logs.

Case id	Timestamp	Tests	Results	...
1001	21/06/2021 18:40	ALT	25	
	21/06/2021 18:40	Potassium	3.6	
	21/06/2021 18:40	Sodium	138	
	22/06/2021 12:06	Glucose	127	

Figure 1. An example laboratory test ordering event log

Process mining has shown significant usage in many fields, especially in the healthcare domain (Rojas et al. 2016). Process mining is adopted to discover common treatment pathway for stroke care (Mans et al. 2008), and for oncology treatment (Kurniati et al. 2018). The healthcare process was improved by identifying the bottleneck in the current patient admission process with the aim to shorten the MRI waiting time (Ganesha et al. 2017). A process mining framework was proposed to detect changes in cancer treatment pathways (Kurniati et al. 2019).

However, research in the healthcare domain mainly pays attention to the frequent pathway since many process discovery algorithms rely on frequency of occurrence as a measure of importance. Hence, lots of infrequent relations are treated as noise and discarded when the number of occurrences is below a pre-set threshold (Mannhardt et al. 2017). Nevertheless, numerous infrequent relations are critical in providing us with useful insights into the process.

In many process discovery algorithms, a threshold is pre-defined. If the number of occurrences does not reach the threshold, the relation would likely to be discarded. Hence, it misses critical insights in the process. Certain infrequent relations are also dependent on outcomes of activities that have happened previously. With the wide availability of electrical health records (EHR), we are able to derive some practical value-based conditions that trigger infrequent relations—taking the process in the emergency room transfer in Figure 2 as an example; the primary pathway is Emergency Department (ED) registration, admission, transfer to relevant care unit and, discharge. Though, albeit few, there are also infrequent traces such as after performing some laboratory tests, some patients are directly discharged. In a common process mining algorithm such as Heuristics Miner (Weijters and Ribeiro 2011), the lab test→discharge relation will be ignored. However, this relation is due to the patient's laboratory test results being normal, meaning this patient is no longer classified as an emergency, so can be discharged from the ED. That is to say, the condition "lab test results normal" triggers the infrequent relation. This infrequent relation should not be discarded, as it provides us with useful information about the process.

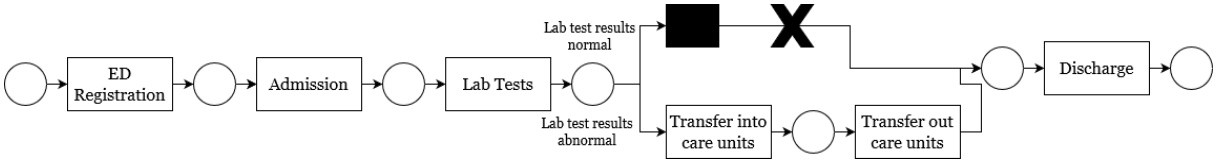

Figure 2. A simplified emergency room transfer process

In this paper, we aim to discover value-based conditions and preserve legitimate infrequent relations in the healthcare process. Specifically, the laboratory testing process is utilised, since laboratory testing is critical in a patient's treatment journey and consists of a large number of different tests. The ordering of

these tests can be complex, as they are usually ordered in sets. It would be helpful if we could identify tests infrequently triggered by other laboratory tests based on their valued-based results. We adopt both the L* life- cycle and Data-aware Heuristic Miner (DHM) as tools. We also extract our own laboratory test ordering event logs using the publicly available MIMIC-III dataset.

This paper is organised as follows. Section 2 introduces the essential background and related work; Section 3 introduces concepts used throughout the paper; Section 4 proposes the methodologies and the dataset; Section 5 presents the results; and Section 6 discusses the outcomes. Section 7 concludes the paper.

2 Background

2.1 Process Mining

Process mining is a newly developed technology that connects computation intelligence, data mining, process modelling, and analysis approaches. The basic scope of process mining is to discover, monitor, and improve processes by extracting knowledge from collected event logs that are currently in the business information system (Van Der Aalst 2016). With process mining, a model is produced by analysing the event logs from a database such as EHRs. This model makes it possible to compare different processes, improve performance, and predict potential outcomes.

There are three different stages—process discovery, conformance checking, and process enhancement (Ferreira 2017). The most crucial stage is process discovery, which involves generating accurate representative models from the event log. Conformance checking evaluates whether the discovered process model conforms with the event log based on the criteria of fitness, generalisation, precision, and simplicity (Carmona et al. 2018). Enhancement refers to improving the real-life process based on the discovered process model, for example, identifying process bottlenecks and simulating potential changes in the process.

As a result, most process mining techniques aim to discover the main pathway of the process to avoid overfitting (Augusto et al. 2018). Common discovery algorithms such as the α-miner and the Heuristics Miner (Weijters and Ribeiro 2011) filter out infrequent relations when mining process models. However, behaviours exist that are infrequent but important under specific contexts. Researchers developed a tool to consider the resources of activities during the process discovery stage to discover such behaviours (Mannhardt et al. 2017). Unfortunately, it has hardly been applied to the healthcare domain.

2.2 Process Mining in Healthcare

In recent years, there has been an increasing number of applications of process mining in the field of healthcare. A large group of applications are based on Fuzzy Miner algorithm (Pelekis et al. 2005) and its commercial application named Disco (Günther and Rozinat 2012), as they are easy to use. These tools are also ideal for discovering casual relations between activities. For example, researchers use Disco to identify the process of the ED and verify whether certain clinical guidelines are satisfied (Alvarez et al. 2018). Since Disco and Fuzzy Miner use unified noise thresholds for all the behaviours, such infrequent but important relations will be ignored. Researchers combined the Disco tool with data mining techniques to estimate waiting time in the ED (Benevento et al. 2019).

Another group of researchers aimed to extract a model with semantics (i.e., BPMN, Petri-nets, casual-nets) from healthcare data. For example, patient transport data was extracted from the MIMIC-III dataset to analyse how patients are transferred in and out of ICU by discovering process models using the inductive miner (Kurniati et al. 2018). Studies consider both the order and resource of activities when discovering process models to study the allocation of resources of health services and medical facilities (Prokofyeva et al. 2020; Stefanini et al. 2020). Although some studies have applied process mining in the healthcare (Rojas et al. 2016), few of them focus on the critical infrequent relations among activities in healthcare.

3 Preliminaries

Before proposing our novel ideas, this section introduces some concepts needed in Section 4.

3.1 Event log

An event log is defined as L = (E,A,V,N,#,T). E is the set of unique event identifiers; A is the sets of activities; V is the sets of data resources; N is the sets of numerical resource names; #: E→(N→V) is a

function that obtains data resources recorded for an event e ∈ E—for example, $\#_{ac}(e)$ gets the activity name for an event, $\#_{n}(e)$ gets the numerical data resources for an event; and $T \subseteq E^{*}$ is the set of traces over E. A trace $t \in T$ records the sequence of events of a process instance. Each event only occurs once in a single trace.

3.2 Directly-follows Relation

$a >_{W} b$ holds if there is a trace $t \in T$ where $t(i) = e_1$ and $t(i+1) = e_2$ and $\#_{ac}(e_1) = a$ and $\#_{ac}(e_2) = b$.

3.3 Data-driven Heuristic Miner

Heuristic Miner (HM) is a well-known process discovery algorithm (Weijters et al. 2006). It takes the event log as input and generates a Petri net that represents the process's main behaviours. HM relies on occurrence frequencies to determine which relation to include in the process model. For instance, if the frequency is below the threshold, the relation between two activities will be discarded. HM reduces the negative impact of noise and incorrect data, while some important but rare relations would also be dropped. The Data-driven Heuristic Miner (DHM) algorithm improves the HM by revealing and preserving conditional infrequent behaviours from the event log (Mannhardt et al. 2017). DHM is perfectly suitable in this study, as we would like to discover infrequent relations between laboratory tests along with conditions that may trigger them. Figure 3 illustrates the overall steps for DHM. There are three main parameters in the algorithm, these are:

- θ_obs, which controls the relative frequency of relations.
- θ_dep, which controls the frequency of relations.
- θ_con, which controls the quality of conditions.

First, the DHM tries to calculate the frequency of every relation by adopting the ideas from the traditional HM. Then, it identifies infrequent relations through θ_dep and θ_obs; if the dependency of the relations is smaller than θ_dep, or the relative frequency is smaller than θ_obs. Such relations would be considered as infrequent relations. The next step is to discover conditions which may trigger infrequent relations. DHM builds training instances for every infrequent relation and deploys a decision tree to train the instances using the resources of these events. Then the DHM provides two ways to evaluate conditions. One adopts Cohen's kappa (Cohen 1960) to assess conditions. Those conditions that score lower than θ_con will be discarded. The other method to evaluate conditions relies on a relations' frequency under specific conditions—whether they exceed θ_dep in event logs. DHM returns a C-net graph as the final output, where infrequent relations will be marked in different colors.

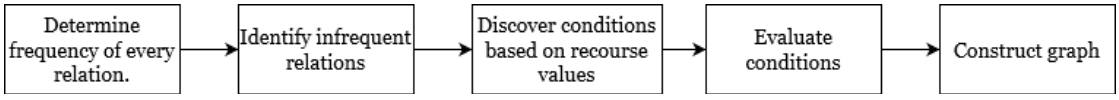

Figure 3. Steps of the DHM algorithm

4 Methodology

This exploratory study aims to discover value-based conditions and preserve useful infrequent relations for standard lab tests in the MIMIC-III dataset. We modify and apply the L* life-cycle methodology (Van Der Aalst 2016), the Data-driven Heuristic Miner (DHM) (Mannhardt et al. 2017) , and propose a novel data preprocessing approach to achieve the goal.

4.1 L* Life-cycle Methodology

The modified L* life-cycle methodology (shown in Figure 4) contains four stages. Stage 0 is to plan and justify, which is the same as the first stage in the original L* methodology. This stage involves planning the research and proposing meaningful research tasks. Stage 1 is extracting and preprocessing, which consists of extracting the data needed from the MIMIC-III database and essential preprocessing steps being applied to the data, as the extracted data is not naturally designed for process mining tasks. Stage 2 is to identify value-based conditional infrequent relations by adopting the DHM algorithm. Stage 3 is evaluation, which preserves useful and meaningful value-based conditions and infrequent relations through specific criteria.

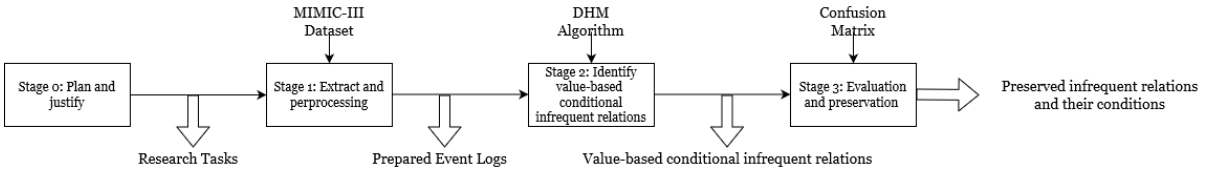

*Figure 4. Overview of L * life-cycle methodology*

4.2 MIMIC-III Dataset

The dataset used for the application in this paper is Medical Information Mart for Intensive Care (MIMIC-III), which comprises EHR information relating to de-identified patients admitted to the critical care unit at a large tertiary care hospital (Johnson et al. 2016). Currently, the latest version is MIMIC-III v1.4, which was released in September 2016 and applied in the present. Meanwhile, besides activity names, MIMIC-III also contains other attributes for these activities, such as the results of the laboratory tests (Johnson et al. 2016), which makes it possible to apply it in the tool to discover those infrequent but important conditional relations between activities (Mannhardt et al. 2017). Lots of studies have utilised the MIMIC-III dataset as the primary data source, such as those predicting ICU readmission (Lin et al. 2019) and mortality rate (Mandalapu et al. 2019).

There are over 25 tables in the MIMIC-III dataset, where 16 are suitable for process mining as they have timestamp information. In this study, we used five tables in data extraction. The d_lab_items table describes the laboratory tests in the dataset, which also serves as a dictionary, as all laboratory tests patient go through in treatment process are recorded as numerical IDs instead of specific names. The labevents table contains the laboratory test measurements for each patient. It is worth noting that two timestamps are available for each test record. One is “charttime” and the other is “storetime”. In this paper, we utilise “charttime” as the timestamp for each test in the event log as this is the best match to the actual time of measurement (Johnson et al. 2016). The d_icd_diagnoses table is also a dictionary, with all diseases and their ICD-9 codes (Cassel and Vladeck 1996). The diagnoses_icd table has the diagnoses from each patient’s each admission to the hospital, and is coded using the ICD system. The patients table contains information about each patient, such as sex and date of birth.

4.3 Other Tools

The tools used in this study include Postgres SQL, Python and ProM. Postgres SQL is used to access and extract data from the MIMIC-III dataset. Python is used for data preprocessing and evaluation. ProM is a software that collects different process mining algorithms, ranging from process discovery to conformance checking (Van Dongen et al. 2005). We adopt the DHM algorithm which is implemented in the ProM in this study,

5 Results

This section presents our results according to stages described in the L*life-cycle methodology in Section 4.

5.1 Stage 0: Plan and Justify

In this study, we present the following three research tasks to guide our research.

- Identify infrequent relations between laboratory tests in different diseases.
- Identify the value-based conditions that can trigger these infrequent relations.
- Preserve useful and precise value-based conditions.

5.2 Stage 1: Extract and Preprocessing

We selected 12 common laboratory tests mentioned in (Houben et al. 2010), along with their numerical IDs in the MIMIC-III dataset, which are summarised in Table 1. Then, we extracted the patients’ laboratory test records, which contain one or more of those common tests. This operation returned a considerable dataset, which has more than 53,132 traces and 1,781,458 events. We also selected five diseases using the ICD-9 code (Cassel and Vladeck 1996), which are summarised in Table 2. Additionally, the dataset of each disease is divided into two separate datasets, one contains female patients, and the other one contains male patients for the purpose of evaluation.

Item ID	Description

1	50861	Alanine Aminotransferase (ALT)
2	50907	Total Cholesterol
3	51080	Creatinine Clearance
4	50889	C-Reactive Protein
5	51288	Sedimentation Rate
6	50924	Ferritin
7	50927	Gamma Glutamyl transferase
8	50809	Glucose
9	50811	Haemoglobin
10	50983	Sodium
11	50971	Potassium
12	50993	Thyroid Stimulating Hormone

Table 1. Summary of laboratory tests

Three different preprocessing steps were applied to the extracted data, including changing the data format, cleaning, and aggregation. Since those conditions were likely to be discovered based on the resource values (e.g., laboratory test results) from previous events, we must consider all of the laboratory test results and find a way to add these results to the event log as resources. To do so, we manually added a column to store results for each test. Each column is named in the form of 'test name$_{result}$'. For example, if the event is ALT test, then the result is stored at the ALT_{result} column and all other columns for this event are left blank. In total, 12 columns were added, including ALT_{result}, $Sodium_{result}$, for example. Thus, each test has its attribute, and we can distinguish which test results can trigger infrequent relations. An example event log is shown in Figure 1.

We also identified some traces may not be appropriate to use in this study. We conclude the common issues below. These traces were removed to avoid producing unwanted and unrealistic relations.

- Traces containing only a single event,
- Traces containing only a single kind of test,
- Traces containing laboratory test results which have non-numerical or ambiguous value, such as 'greater than 2000', 'error', 'smaller than 10' etc.

During our experiment, we observed an interesting phenomenon in the labevents table, where many test events have the same timestamp, meaning these tests are all happening simultaneously. This is exemplified in Figure 1. So, we were unable to identify the actual order these tests were done in. Given tests are usually ordered in sets by doctors, we propose grouping the same timestamp tests together within order sets, as seen in Figure 1.

This approach is described in Algorithm 1 and implemented as a Python program. The method takes the original log, user-defined pattern length, and frequency as inputs. A pattern is a set of tests that occur at the same time within a trace. The length of the pattern calculates how many tests in the pattern. Frequency represents the times of occurrence for a specific pattern. First, we identified patterns within each trace using pattern length. Each time a new pattern is found, it is added to the set S (Lines 1–2). For example, I is set to two, which means if two or more tests happened simultaneously, we consider it a pattern. Next, we went through the pattern set and counted how many times each pattern appeared in the set. A dictionary is created with the pattern as the key and its frequency as the value. This can avoid duplicate patterns as the dictionary in Python does not allow for key duplications (Lines 3–5). If the frequency of a pattern is smaller than the given threshold, which is the pattern frequency, the pattern will be removed from the dictionary (Lines 6–8). Lastly, we went through each trace, and merged each occurrence of the pattern into a test set by combining tests using "+" symbol, while maintaining all test results as resources (Lines 9–11). For instance, a test set can be "Ferritin+Potassium+Sodium". We grouped tests that typically occur simultaneously with high occurrence frequency into test sets and returned the preprocessed event log. An example of a preprocessed event log is shown in Figure 5.

ALGORITHM 1: Merge the same timestamp tests into test sets

Input: original log L= (E,A,V,N,#,T), pattern length I, and pattern frequency f

Output: Merged event logs L_{new}

1 foreach $t_i \in T$ do

2 | Pattern Set S←S+IdentifyPattern(t_i, I)

```
foreach s∈S do
    i←counttimes(s)
    Pattern Dictionary D ←(s,i)
foreach (d,i)∈D do
    if i <f then
        D ←D-(d,i)
foreach t_i∈T do
    t_i←AggregatePattern(D,t_i)
    L_new←L_new+t_i
return L_new
```

Case id	Timestamp	Tests	ALT_{result}	$Potassium_{result}$	$Sodium_{result}$	$Glucose_{result}$
1001	21/06/2021 18:40	ALT+Potassium +Sodium	25	3.6	138	
	22/06/2021 12:06	Glucose				127

Figure 5. An example merged laboratory tests ordering event log

	ICD_9 Code	Long Title
1	4019	Unspecified essential hypertension
2	4280	Congestive heart failure, unspecified
3	5849	Acute kidney failure, unspecified
4	41401	Coronary atherosclerosis of native coronary artery
5	42731	Atrial fibrillation

Table 2. Summary of Diseases

	Name	#Traces	#Events	#Activities	Average Length	Maximum Length
1	Icd_42731_female	5381	111939	23	21	225
2	Icd_42731_male	7279	169773	25	23	348
3	Icd_4019_female	9170	147558	27	16	290
4	Icd_4019_male	11161	193266	26	17	571
5	Icd_41401_female	4250	80792	21	19	244
6	Icd_41401_male	7951	153178	23	19	305
7	Icd_5849_female	3894	83708	22	21	236
8	Icd_5849_male	5142	118856	24	23	571
9	Icd_4280_female	6007	120775	25	20	280
10	Icd_4280_male	6935	158164	26	23	478

Table 3. Summary of datasets

5.3 Stage 2: Identify Infrequent relations

In Stage 2, we applied the 'Interactive Data-aware Heuristic Miner (iDHM)' in the ProM, which implements the DHM algorithm (Mannhardt et al. 2017). In this study, we adopted the default value for θ_obs which is 0.1. The default value 0.9 for θ_dep is too limited for the lab tests, as patient's journeys are of high variance. θ_dep is adjusted to 0.8 in our study. Figure 6 provides an example on how the DHM utilises test results to discover value-based conditions for infrequent relations. Suppose the infrequent relation we find is Test set A→Test B. Test set A consists of three different tests and their results. We also have a frequent relation: Test set C→Test A. Test set C contains two tests and their results. When DHM tries to discover the conditions of infrequent relations, it will take all tests' results before Test B (i.e., result A, B, C, and D) into consideration instead of only the one from its direct predecessor. However, it is notable that there are two 'Results C' in Figure 6, the DHM will only consider the latest one, which is the one from Test set A. Then the DHM evaluates conditions based on θ_con,

where the default value 0.5 is retained. In the study, iDHM was set to consider all the laboratory test results when mining value-based conditions.

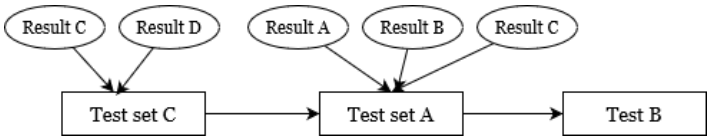

Figure 6. An example of how DHM discover conditions based on attributes

5.4 Stage 3: Evaluation and Preservation

We evaluated the results in Stage 2 from several perspectives. First, we excluded relations which contain the 'artificial start' or 'artificial end', because the DHM adds these two activities to construct a completed C-net. It is unclear why this occurs. For example, the Sodium→Artificial_end relation will be discarded, which is represented as 'Discard (1)' in Table 4. Since we are interested in conditions derived from the previous test results, which may trigger another test, all relations with no such conditions will be excluded, written as 'Discard (2)'. For example, the Sodium→Glucose relation with the condition $\text{Potassium}_{\text{result}}$>12 mEq/L is ignored because the condition is irrelevant to the sodium test. Besides, we also measured the accuracy of each condition using a separate dataset. We separated each disease dataset into two, according to gender and randomly picked one as the training dataset, which became the input for the DHM; the other serves as the evaluation dataset. The concept of the confusion matrix is implemented in this stage for evaluation (Stehman 1997). The ALT → Potassium relation with the condition $\text{ALT}_{\text{result}}$<21 IU/L serves as an example to illustrate the confusion matrix. True-positive (TP) is the number of relations that happen under the condition we find; i.e., a patient has the Potassium test directly after the ALT test with the $\text{ALT}_{\text{result}}$<21 IU/L. True-negative (TN) is the number of relations that do not occur if the condition is not met; i.e., a patient with $\text{ALT}_{\text{result}}$≥21 IU/L and the potassium test does not happen. False-positive (FP) is the number of relations that do not happen despite the condition being met, i.e., a patient with $\text{ALT}_{\text{result}}$<21 IU/L, but no potassium test directly after the ALT test. False-negative (FN) is the number of relations that happen despite the condition not being met; i.e., a patient takes the potassium test directly after the ALT test, but their $\text{ALT}_{\text{result}}$≥21 IU/L. The equation for calculating accuracy is provided in Equation 1. The infrequent relations, along with their value-based conditions, can be preserved if the user defined accuracy threshold is satisfied.

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN) \qquad \textit{Equation 1}$$

Those conditions with the accuracy below the accuracy threshold are excluded, represented as 'Discard (3)' in Table 4. The threshold is set to 0.7 in the study.

	Reason
Discard (1)	Relations contain artificial start or end.
Discard (2)	Conditions are not fully derived from previous test results.
Discard (3)	Condition's accuracy below certain threshold.

Table 4. Summary of discard reasons

6 Discussion

In total, out of five diseases, we successfully identified 24 infrequent relations. Among them, 12 relations were discarded in Stage 3. Most of the infrequent relations we found occurred less than 10 times in event logs. We regard the remaining 12 as useful relations with value-based conditions. Among the 12 removed relations, five were removed due to an accuracy below 0.7 and four were removed because their relations included an artificial start or end. On average, the preserved relations achieved 87% accuracy, while the highest accuracy was 97%, and the lowest was 70%.

By applying the methodology described in Section 3, the tasks proposed in Stage 0 can easily be achieved. Regarding the first task, we randomly picked a female or male dataset from each disease to mine the infrequent relations; the results are summarised Table 5. We picked the female dataset using atrial fibrillation (ICD-9: 42731) as an example and found six infrequent relations. One infrequent relation can occur after patients go through a set of tests, which include ALT, ferritin, potassium and

sodium; under some circumstances, they may directly have a haemoglobin test. For the second task, conditions for each infrequent relation are also presented in Table 5. For example, the condition triggers the first relation in atrial fibrillation is $Ferritin_{result}$>335 ng/ml . We can interpret the condition in this way: the relation will not frequently happen and is considered as noise. However, suppose the ferritin result for a patient is greater than 355 ng/ml during the sets of tests, which include ALT, ferritin, potassium and sodium—then the patient is very likely to be asked to perform another haemoglobin test afterwards. For task 3, we evaluated conditions according to the criteria in Stage 3. We removed conditions that did not satisfy requirements and preserved the rest. For the preserved relation, the accuracy is also displayed in Table 5. These provided us with an insight into infrequent laboratory test relations and their value-based conditions in different diseases.

However, we also noticed that some conditions have low accuracy. Several reasons may explain this. First, infrequent conditional relations may not be the same in various datasets because we separated datasets of each disease according to gender. So the difference may exist between them naturally. Of note too, various admission types exist in the MIMIC-III dataset, such as elective and emergency. Understandably, the ordering of laboratory tests for different admission types are varied, which may be a future research area.

Infrequent Relation	Value-based Condition	Keep/Discard
Training dataset: Icd_4280_female	Evaluation dataset: Icd_4280_male	
ALT+Cholesterol+Potassium+Sodium⟶ Hemoglobin	ALT_{result}<=21 IU/L	Discard (3)
ALT+Cholesterol+Patassium+Sodium⟶ Hemoglobin+Glucose	ALT_{result}<=62 IU/L	Discard (3)
Hemoglobin⟶Ferritin+Potassium+Sodium+ Thyroid Stimulating Hormone	$Hemoglobin_{result}$>12.2 g/dL	Keep (0.85)
Potassium+Sodium+ThyroidStimulating Hormone ⟶Sodium	$Sodium_{result}$<=129 mEq/L	Keep (0.97)
Training dataset: Icd_5849_male	Evaluation dataset: Icd_5849_female	
Cholesterol+Potassium+Sodium⟶Ferritin+Potassium+Sodium	($Sodium_{result}$>145mEq/L)&($cholesterol_{result}$<169 mg/dL)	Keep (0.94)
Cholesterol+Potassium+Sodium⟶Cholesterol+Potassium+Sodium	$cholesterol_{result}$>169 mg/dL	Keep (0.70)
ALT+Potassium+Sodium+Thyroid Stimulating Hormone⟶ALT+Ferritin+Potassium+Sodium	$Sodium_{result}$<=134 mEq/L	Keep (0.81)
Ferritin+Potassium+Sodium⟶ARTIFICIAL_ END		Discard (1)
Training dataset: Icd_41401_male	Evaluation dataset: Icd_41401_female	
ALT+Cholesterol+Potassium+Sodium ⟶ALT+Cholesterol+Potassium+Sodium	($Potassium_{result}$<=3.4 mEq/L)&($cholesterol_{result}$)<190mg /dL)	Keep (0.91)
ALT+Ferritin+Potassium+Sodium⟶ALT+Ferritin+Potassium+Sodium	$Sodium_{result}$<=131 mEq/L	Keep (0.93)
Sodium⟶Potassium+Sodium+Thyroid Stimulating Hormone	($Potassium_{result}$>4.6 mEq/L)&($Sodium_{result}$<=129 mEq/L)	Discard (2)
Training dataset: Icd_4019_female	Evaluation dataset: Icd_4019_male	
Potassium+SedimentationRate+Sodium⟶C-Reactive Protein	ESR_{result}>61.0 mm/hr	Discard (3)
Potassium+SedimentationRate+Sodium⟶Potassium	($Sodium_{result}$>140mEq/L)&(ESR_{result}<=36)mm/hr	Keep (0.90)

ThyroidStimulatingHormone⟶ALT+Potassium+Sodium+Thyroid Stimulating Hormone	$Sodium_{result}$<=135 mEq/L	Discard (2)
ALT+Cholesterol+Potassium+Sodium⟶Glucose+Hemoglobin	$Sodium_{result}$>142 mEq/L	Keep (0.82)
Potassium+SedimentationRate+Sodium⟶Potassium+SedimentationRate+Sodium	ESR_{result}>65 mm/hr	Discard (3)
Hemoglobin⟶Hemoglobin	$Potassium_{result}$<=4.4 mEq/L	Discard (2)
ALT+Potassium⟶ARTIFICIAL_END		Discard (1)
Training dataset: Icd_42731_female	Evaluation dataset: Icd_42731_male	
ALT+Ferritin+Potassium+Sodium⟶Hemoglobin	$Ferritin_{result}$>335 ng/ml	Discard (3)
ALT+Ferritin+Potassium+Sodium⟶ARTIFICIAL_END		Discard (1)
Ferritin+Potassium+Sodium⟶Sodium	$Sodium_{result}$<=128 mEq/L	Keep (0.98)
ALT+Ferritin+Potassium+Sodium⟶Cholesterol+Potassium+Sodium	($Ferritin_{result}$>149ng/ml)&($Ferritin_{result}$<311 ng/ml)	Keep (0.79)
ThyroidStimulatingHormone⟶ARTIFICIAL_END		Discard (1)
ALT+Ferritin+Potassium+Sodium⟶Glucose+Hemoglobin	$Ferritin_{result}$<=114 ng/ml	Keep (0.85)

Table 5. Summary of results

7 Conclusion

Process mining applications in healthcare rely on frequency as the important criterion to preserve relations. Hence, some rare but important relations would be discarded, leading to the loss of information. This paper successfully implemented a value-based condition discovery pipeline on a publicly available hospital EHR dataset. We preserved legitimate infrequent relations by successfully modifying and applying the L* life-cycle methodology and Data-aware Heuristic Miner (DHM) to the laboratory test ordering process. The results and evaluations show that we can successfully preserve useful value-based conditions for infrequent relations discovered through the pipeline. Future work includes investigating the differences between each disease and utilising domain knowledge to evaluate infrequent relations along with their conditions.

8 References

Alvarez, C., Rojas, E., Arias, M., Munoz-Gama, J., Sepúlveda, M., Herskovic, V., and Capurro, D. 2018. "Discovering Role Interaction Models in the Emergency Room Using Process Mining," *Journal of biomedical informatics* (78), pp. 60-77.

Augusto, A., Conforti, R., Dumas, M., La Rosa, M., Maggi, F. M., Marrella, A., Mecella, M., and Soo, A. 2018. "Automated Discovery of Process Models from Event Logs: Review and Benchmark," *IEEE transactions on knowledge and data engineering* (31:4), pp. 686-705.

Benevento, E., Aloini, D., Squicciarini, N., Dulmin, R., and Mininno, V. 2019. "Queue-Based Features for Dynamic Waiting Time Prediction in Emergency Department," *Measuring Business Excellence*).

Carmona, J., van Dongen, B., Solti, A., and Weidlich, M. 2018. *Conformance Checking*. Springer.

Cassel, C. K., and Vladeck, B. C. 1996. "Icd-9 Code for Palliative or Terminal Care." Mass Medical Soc.

Cohen, J. 1960. "A Coefficient of Agreement for Nominal Scales," *Educational and psychological measurement* (20:1), pp. 37-46.

Ferreira, D. R. 2017. "A Primer on Process Mining," *Practical Skills with Python ang Graphviz*).

Ganesha, K., Supriya, K., and Soundarya, M. 2017. "Analysing the Waiting Time of Patients in Hospital by Applying Heuristics Process Miner," *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*: IEEE, pp. 500-505.

Günther, C. W., and Rozinat, A. 2012. "Disco: Discover Your Processes," *BPM (Demos)* (940), pp. 40-44.
Houben, P. H., Winkens, R. A., Van der Weijden, T., Vossen, R. C., Naus, A. J., and Grol, R. P. 2010. "Reasons for Ordering Laboratory Tests and Relationship with Frequency of Abnormal Results," *Scandinavian journal of primary health care* (28:1), pp. 18-23.
Johnson, A. E., Pollard, T. J., Shen, L., Li-Wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. 2016. "Mimic-Iii, a Freely Accessible Critical Care Database," *Scientific data* (3:1), pp. 1-9.
Kurniati, A. P., Hall, G., Hogg, D., and Johnson, O. 2018. "Process Mining in Oncology Using the Mimic-Iii Dataset," *Journal of Physics: Conference Series*: IOP Publishing, p. 012008.
Kurniati, A. P., McInerney, C., Zucker, K., Hall, G., Hogg, D., and Johnson, O. 2019. "A Multi-Level Approach for Identifying Process Change in Cancer Pathways," *International Conference on Business Process Management*: Springer, pp. 595-607.
Lin, Y.-W., Zhou, Y., Faghri, F., Shaw, M. J., and Campbell, R. H. 2019. "Analysis and Prediction of Unplanned Intensive Care Unit Readmission Using Recurrent Neural Networks with Long Short-Term Memory," *PloS one* (14:7), p. e0218942.
Mandalapu, V., Ghaemmaghami, B., Mitchell, R., and Gong, J. 2019. "Understanding the Relationship between Healthcare Processes and in-Hospital Weekend Mortality Using Mimic Iii," *Smart Health* (14), p. 100084.
Mannhardt, F., de Leoni, M., Reijers, H. A., and van der Aalst, W. M. 2017. "Data-Driven Process Discovery-Revealing Conditional Infrequent Behavior from Event Logs," *International conference on advanced information systems engineering*: Springer, pp. 545-560.
Mans, R., Schonenberg, H., Leonardi, G., Panzarasa, S., Cavallini, A., Quaglini, S., and Van Der Aalst, W. 2008. "Process Mining Techniques: An Application to Stroke Care," *MIE*, pp. 573-578.
Mans, R. S., Schonenberg, M., Song, M., Van der Aalst, W., and Bakker, P. 2015. "Process Mining in Healthcare," *International Conference on Health Informatics (HEALTHINF'08)*, pp. 118-125.
Pelekis, N., Theodoulikis, B., Kopanakis, I., and Theodoridis, Y. 2005. "Fuzzy Miner: Extracting Fuzzy Rules from Numerical Patterns," *International Journal of Data Warehousing and Mining (IJDWM)* (1:1), pp. 57-81.
Prokofyeva, E. S., Maltseva, S. V., Fomichev, N. Y., and Kudryashov, A. G. 2020. "Data-Driven Approach to Patient Flow Management and Resource Utilization in Urban Medical Facilities," *2020 IEEE 22nd Conference on Business Informatics (CBI)*: IEEE, pp. 71-77.
Rojas, E., Munoz-Gama, J., Sepúlveda, M., and Capurro, D. 2016. "Process Mining in Healthcare: A Literature Review," *Journal of biomedical informatics* (61), pp. 224-236.
Stefanini, A., Aloini, D., Benevento, E., Dulmin, R., and Mininno, V. 2020. "A Data-Driven Methodology for Supporting Resource Planning of Health Services," *Socio-Economic Planning Sciences* (70), p. 100744.
Stehman, S. V. 1997. "Selecting and Interpreting Measures of Thematic Classification Accuracy," *Remote sensing of Environment* (62:1), pp. 77-89.
Van Der Aalst, W. 2016. "Data Science in Action," in *Process Mining*. Springer, pp. 3-23.
Van Dongen, B. F., de Medeiros, A. K. A., Verbeek, H., Weijters, A., and van Der Aalst, W. M. 2005. "The Prom Framework: A New Era in Process Mining Tool Support," *International conference on application and theory of petri nets*: Springer, pp. 444-454.
Weijters, A., and Ribeiro, J. T. S. 2011. "Flexible Heuristics Miner (Fhm)," *2011 IEEE symposium on computational intelligence and data mining (CIDM)*: IEEE, pp. 310-317.
Weijters, A., van Der Aalst, W. M., and De Medeiros, A. A. 2006. "Process Mining with the Heuristics Miner-Algorithm," *Technische Universiteit Eindhoven, Tech. Rep. WP* (166), pp. 1-34.

Copyright