Discovering Break Behaviours in Process Mining: An Application to Discover Treatment Pathways in ICU of Patients with Acute Coronary Syndrome

Qifan Chen*1[0000−0003−1068−6408], Yang Lu*1[0000−0002−9002−8650], Charmaine S. Tam2[0000−0002−4034−9923], and Simon K. Poon1[0000−0003−2726−9109]

1 School of Computer Science, The University of Sydney, Sydney, Australia
{qifan.chen,yang.lu,simon.poon}@sydney.edu.au
2 Northern Clinical School, University of Sydney, Sydney, Australia
charmaine.tam@sydney.edu.au

Abstract. The inductive miner (IM) can guarantee to return structured process models, but the process behaviours that process trees can represent are limited. Loops in process trees can only be exited after the execution of the "body" part. However, in some cases, it is possible to break a loop structure in the "redo" part. This paper proposes an extension to the process tree notation and the IM to discover and represent break behaviours. We present a case study using a healthcare event log to explore Acute Coronary Syndrome (ACS) patients' treatment pathways, especially discharge behaviours from ICU, to demonstrate the usability of the proposed approach in real-life. We find that treatment pathways in ICU are routine behaviour, while discharges from ICU are break behaviours. The results show that we can successfully discover break behaviours and obtain the structured and understandable process model with satisfactory fitness, precision and simplicity.

Keywords: Process Mining · Inductive Miner · Process Trees · Healthcare Process Discovery.

1 Introduction

Process discovery algorithms are techniques that construct process models automatically from recorded data. Unlike many machine learning techniques, it has been discussed that process mining methods should return human-understandable results [3]. However, many existing process discovery algorithms return "spaghetti-like" process models which are hard to understand [3]. The inductive miner (IM) [17] is one of the most popular process discovery results that guarantees to return structured process models with high fitness. Although structured models are guaranteed to be discovered, as the direct output is a process tree [17], the possible represented behaviours are limited [20].

* The first two authors contribute equally to the paper.

In process trees, a loop is composed of a “body” part and multiple “redo” parts. A loop should always start and end with the “body” part. For example, in $\circlearrowleft (T_1, T_2)$, the loop should always start with T_1, then choose if it executes T_2 or exits the loop. However, in certain cases, it is possible to exit the loop from the “redo” part (e.g., exiting the loop during the execution of T_2). Such behaviours are called break behaviours which cannot be represented by the process trees or discovered by the IM.

Healthcare process discovery aims to get insights into how healthcare processes are executed and identify opportunities for improving services [23]. Tremendous efforts have been put into discovering various healthcare processes [9,21,22,23,26]. The Intensive Care Unit (ICU) is expensive, and its cost increases yearly. Carefully deciding on the discharging of patients from ICU is thus a critical issue to guarantee efficient treatments have occurred. Hence, the discovery of treatment processes in ICU can help domain experts better understand how patients are treated and improve the quality of medical services.

We present a motivation example (shown in Fig 1) to demonstrate break behaviours in ICU treatment. The ICU treatment pathway is a routine loop behaviour that involves continued monitoring of vital measurements and repetitive orders several times a day (e.g., laboratory tests and medications) until discharge to normal wards. Patients are admitted to ICU and nurses start to monitor the vital measurements. Blood tests, followed by medications may be ordered for patients as requested by doctors. After staying in ICU for some time (i.e., executing the routine loop several times), patients can be discharged from ICU through three different ways. Typically, patients should be discharged if their vital measurements are normal, which is discharge pathway one in Fig 1. Nevertheless, patients can be discharged after further ordering blood tests (discharge pathway two in Fig 1) and medications (discharge pathway three in Fig 1). Such discharge pathways are break behaviours (e.g., a break from the “redo” part of the routine treatment loop).

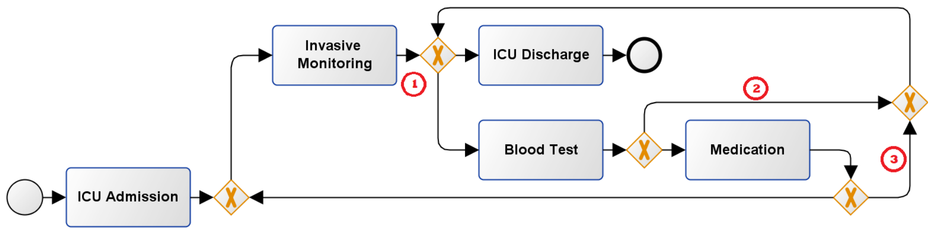


Fig. 1. A motivation example of a ICU treatment process

This paper proposes an extension to the process tree notation and the IM to discover and represent break behaviours. A case study is presented using a real-life healthcare event log to demonstrate the usability of the proposed approach. We aim to discover the treatment pathways in ICU of patients with

Acute Coronary Syndrome (ACS). Specifically, we plan to investigate the discharge behaviours from ICU, which can be regarded as break behaviours. The results show that we can obtain the structured and straightforward process model with satisfactory fitness, precision and simplicity.

The paper is structured as follows: Section 2 discusses the background. Section 3 explains the main approach. A real-life case study is presented in Sect. 4. Section 5 concludes the paper.

2 Background

2.1 Process Discovery Algorithms

The IMs [13,14,15,16,17] are a family of process discovery algorithms that apply the divide-and-conquer method to discover process trees. As process trees are discovered, the discovered process will always be structured and understandable. However, due to the limitation of process trees, certain behaviours are inherently harder to be represented and discovered. "Flower model" is often the result when the input log is complex and unable to be adequately represented by the process tree. Some extensions are proposed to discover more behaviours using the IM (e.g., recursive behaviours [12], switch behaviours [20], and cancellation behaviours [11]).

For those process discovery algorithms which can discover break behaviours in loops, like the alpha miner [1], the heuristics miner [25] and the split miner (SM) [4], structured process models cannot be guaranteed, instead "spaghetti like" process models are often returned [3].

2.2 Process Discovery in Healthcare

With the rapid development of process mining, healthcare process discovery has recently drawn even more attention [23]. [9,22] aim to analyse the trajectories of patients in hospitals, while [21,26] model the workflows in outpatient departments. Patient pathways in emergency departments are discovered in [2,8]. However, less attention is paid to the treatment process for patients in ICU, even though ICU is critical for patients with severe health conditions. Carefully deciding on the transfer out of patients in ICU is thus essential to ensure that patients receive efficient treatments. Apart from normal vital measurements, patients may need to undertake other examinations or treatments before discharging from ICU. Hence, we aim to discover treatment pathways (especially discharge behaviours for patients with ACS) in ICU. Additionally, due to the distinguishing characteristics of healthcare processes, the existing process discovery algorithms constantly fail to discover structured process models [23]. Therefore, unnecessary difficulties have been added to understanding the discovered process models for domain experts. Interactive process discovery is then proposed to address the issue [5,21]. Unfortunately, such domain knowledge is not always available under given conditions. Hence, we propose an approach to automatically discover structured and understandable process models, especially when complex behaviours (i.e., break behaviours) exist in healthcare event logs.

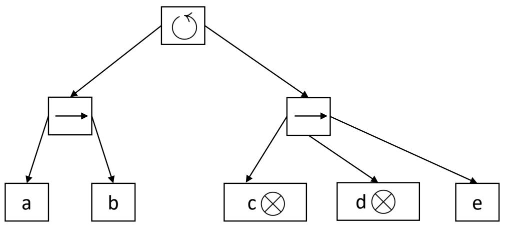
3 Methodology

3.1 The Break Process Tree

To represent break behaviours in process trees, we define the break process tree in this section to represent break behaviours. The break process tree is an extension based on the process tree model described in [16].

Definition 1 (Break Behaviour). *Assume there is a loop process tree* $T = \circlearrowleft (P_1, P_2, ..., P_n)$*, where* L *is its corresponding event log. There is a break behaviour in* T *if there exists an activity* $a_{end} \in End(L)$*,* $a_{end} \in P_i$*,* $1 < i \leqslant n$*.*

Definition 2 (Break Process Tree). *Assume a finite alphabet* A*. A break process tree is a normal process tree with break leaf operators* $a\otimes$*, where* $a \in A$*. Combined with a loop operator* $\circlearrowleft$*, the break leaf node denotes the place where we execute activity* a*, and have an option to exit the loop. Assume there is a loop process tree* $T = \circlearrowleft (P_1, P_2, ..., P_n)$ *A break leaf node must be placed in the redo part of a loop process tree (i.e.,* $a\otimes \in P_i, 1 < i \leqslant n$*).*



Trace	Break Behaviors
ab	NO
abcdeab	NO
abcd	YES
abcdeabcd	YES
abc	YES
abcdeabc	YES

Fig. 2. An example break process tree and its corresponding traces

An example break process tree and its log are presented in Fig. 2. The process tree contains two break leaf operators $c\otimes$, $d\otimes$ that allow exiting the loop on the redo part of the loop process tree. If the process tree does not contain the break leaf operator, it will always start and end with "ab". The break loop operators allow the process tree to exit the loop after the execution of c and d.

3.2 Discovering Break Process Trees

Our method relies on the IM framework to discover process trees with break behaviours. The original loop cut is replaced with the following three steps:

Step 1: Finding Break Cut In the first step, we aim to find a break loop cut. The break loop cut is similar to the original loop cut but allows the exit of loops from the ”redo” part. The definition of the break loop cut is presented in Definition 3.

Definition 3 (Break Loop Cut). *Suppose G is a directly-follows graph of event log L_{break}. A break loop cut is a partially ordered cut $\sum_1, \sum_2, ..., \sum_n$ of G such that:*

1. All start activities are in $\sum_1$: $Start(G) \subset \sum_1$

2. There must be at least one end activity in $\sum_1$: $\exists a \in \sum_1 : a \in End(G)$

3. There are only edges between $\sum_1$ and $\sum_n$: $\forall m \neq n \neq 1 \wedge a \in \sum_m \wedge b \in \sum_n : (a, b) \notin G$

4. If there are edges from $\sum_1$ to $\sum_n$, the sources of all such edges are end activities: $\exists (a, b) \in G \wedge a \in \sum_1 \wedge b \in \sum_n : a \in End(G)$

5. If there are edges from $\sum_n$ to $\sum_1$, the destinations of all such edges are start activities: $\exists (b, a) \in G \wedge b \in \sum_n \wedge a \in \sum_1 : a \in Start(G)$

6. If there is an edge from $\sum_1$ to $\sum_n$, there should be an edge from all end activities in $\sum_1$ to the same destination in $\sum_n$: $\forall a \in End(G) \wedge a \in \sum_1 \wedge b \in \sum_n : (\exists a^{'} \in \sum_1 : (a^{'}, b) \in G) \iff (a, b) \in G$

7. If there is an edge from $\sum_n$ to $\sum_1$, there should be an edge from the same source to all start activities: $\forall a \in Start(G) \wedge b \in \sum_n : (\exists a^{'} \in \sum_1 : (b, a^{'}) \in G) \iff (b, a) \in G$

Step 2: Identifying Break Leaf Nodes Once a break loop cut is identified, there can be two possibilities: 1) a loop process tree with break behaviours is discovered (i.e., $\exists a \in End(G) \wedge a \in \sum_i \wedge i \neq 1$); 2) a loop process tree without break behaviours is discovered (i.e., $\forall a \in End(G) \wedge a \in \sum_1$). we perform Algorithm 1 to locate the break behaviours. If an empty set is returned, a loop process tree without break behaviours is discovered. Otherwise, we mark the activities in BreakLeadNodes as break leaf nodes.

Algorithm 1: Identifying Break Leaf Nodes

```
Input: A break loop cut $\sum_1, \sum_2, ..., \sum_n$ of directly-follows graph $G$
1 BreakLeafNodes = {}
2 for $i$ in $2 ... n$ do
3     for $a$ in $\sum_i$ do
4         if $a \in End(G)$ then
5             BreakLeafNodes.add(a)
6         end
7     end
8 end
Output: BreakLeafNodes
```

Step 3: Splitting Event Logs The same loop cut split function for the original IM framework is applied to split the event log after a break loop cut. However, splitting the event logs directly after the break loop cut can bring extra behaviours into the discovered process model. For instance, in our example in Fig. 2, the activities are partitioned into two groups after the break loop cut: $\{a, b\}$ and $\{c, d, e\}$. The trace $< a, b, c, d >$ is then divided into $< a, b >$ and $< c, d >$, resulting in a process tree shown in Fig. 3, which allows traces such as $< a, b, c, d, a, b, c, d, a, b >$. To solve the problem, we remove all the traces with break behaviours before splitting the event logs (Algorithm 2).

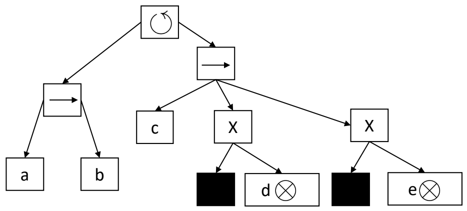


Fig. 3. Break process tree discovered from the log in Fig. 2 if traces with break behaviours are not removed before splitting the log

Algorithm 2: Removing Traces with Break Behaviours

```
Input: A break loop cut $\sum_1, \sum_2, ..., \sum_n$ of directly-follows graph $G$, the event
       log $L_{break}$ of $G$
1 for Trace $t$ in $L_{break}$ do
2     if $End(t) \in \sum_i \wedge i \neq 1$ then
3         $L_{break}.remove(t)$
4     end
5 end
```

A Running Example Finally, a running example shown in Fig. 4 demonstrates the above three steps. The input is the log described in Fig. 2.

4 Case Study

We aim to discover the high-level treatment process in ICU for ACS patients. Specifically, domain experts are interested in how patients are discharged from the ICU. The goals of the case study are:

1. to discover the ACS patients' treatment pathways, especially discharge behaviours from ICU,

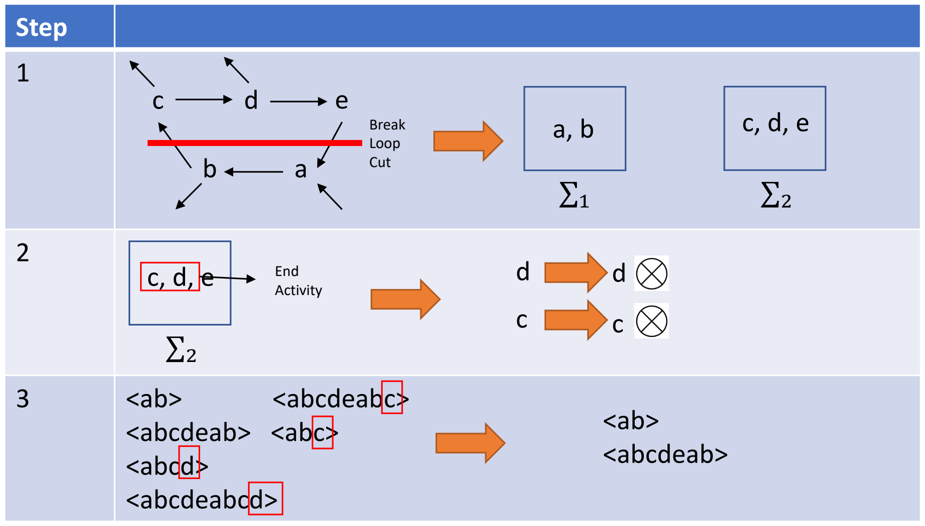


Fig. 4. A running example of our proposed approach based on the example in Fig. 2

2. to quantitative and qualitative evaluate the proposed approach against existing process discovery methods with real-life break behaviours,
3. to gain insights into ICU discharge behaviours for ACS patient.

4.1 Dataset and Event Log Generation

We utilise retrospective data from the EHR extracted between 2013 and 2018 from a single Cerner Millennium Electronic Medical Record domain in Sydney, Australia [7]. The Speed-Extract dataset comprises patients that presented with suspected ACS to facilities in Northern Sydney local health district (LHD) and Central Coast LHD [24]. Ethics and governance approval, including a waiver of informed patient consent, are provided by the Northern Sydney LHD Human Research Ethics Committee for the Speed-Extract dataset [7].

The Speed-Extract dataset consists of 18 tables; the following patient data is provided: Demographic and diagnosis information on patients, triage information after patients have arrived at hospitals, and transfer information between different ward levels. The orders placed during their stays, such as radiology, laboratory tests and procedures, are also provided.

We focus on a particular type of ACS, ST-elevation myocardial infarction (STEMI). STEMI patients are identified using the ICD-10 code (I21.3). We target at patients admitted to emergency wards and directly transferred to ICU. Patients who spent less than 24 hours in hospitals are excluded. Furthermore, patients older than 85 or younger than 40 are excluded, as they have shown to be less informative in the treatment process development [7].

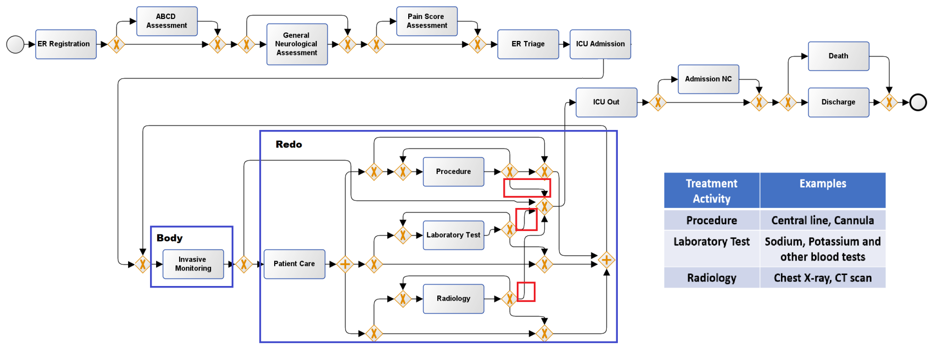
We treat each encounter as a case in the event log, as every encounter represents a unique hospital interaction. In total, 1,582 cases that have 666 variants are included. As behaviours in ICU can be relatively complex without proper abstractions, we categorise the behaviours into five activities: *Invasive Monitoring*, *Patient Care*, *Laboratory Test*, *Radiology* and *Procedure*. We follow [10] to extract frequent laboratory tests performed in ICU. *Radiology* represents the imaging operations in ICU [19], while *Procedure* involves the procedures performed on ICU patients (e.g., breathing assistance with ventilators). Overall, the event log contains events for 15 activities:

– two activities regarding the registration and triage in the emergency ward,
– three activities regarding patient conditions assessments in the triage,
– three activities regarding transfer to normal ward or ICU,
– five activities regarding treatments and measurements in ICU,
– two activities regarding discharge or death.

4.2 Results

Goal 1: ICU Treatment Process Discovery The model discovered by our method is presented in Fig. 5 where the break process tree is translated into a BPMN model. The process starts with patient registration (*ER Registration*) and ends with different outcomes (*Discharge* or *Death*). One trajectory is that patients are directly admitted to ICU after registration. The remaining patients have been assessed before triage. We find that the treatment pathways in ICU are routine (i.e., a loop structure). The treatment pathway starts with invasive monitoring (i.e., “body” part in Fig. 5) for vital measurements [6]. The patients can be discharged from ICU at this point, given that they have received sufficient treatment and their monitoring results are normal. If not, nursing care (*Patient Care*) is conducted afterwards, involving care such as turning the patient in the bed. Patients can be asked to perform several treatments or measurements (i.e., “redo” part in Fig. 5) decided by the ICU team, depending on the monitoring results. Patients can also be discharged during the routine ICU treatment process, given that they have met the discharge criteria [18]. Such discharge behaviours (marked red in Fig. 5) can be considered break behaviours (i.e., a break from the “redo” part of the routine ICU treatment process). In fact, the orders placed in ICU are usually in bulk and made at the beginning of each day. We commonly observe that some orders are cancelled (e.g., discontinued or withdrawn) because patients have met the discharge criteria and been transferred out from ICU. After discharging from ICU, patients can be either discharged from the hospital or admitted to normal wards.

Goal 2: Validating with Existing Process Discovery Methods To compare our method with existing process discovery algorithms, we apply our method, the inductive miner infrequent (IMF) [17], and the SM [4] on the extracted event log. We first apply conformance checking to the three process models. The results are presented in Table.1. Our method and the SM can achieve higher fitness and precision than the IMF. In addition, we adopt size and CFC (the number of branching caused by split gateways) to report the complexity of the process



Treatment Activity	Examples
Procedure	Central line, Cannula
Laboratory Test	Sodium, Potassium and other blood tests
Radiology	Chest X-ray, CT scan

Fig. 5. The process model discovered by the proposed approach.

models. Although our model's size is slightly larger than the SM's, our method achieves a lower CFC.

Table 1. Conformance checking results for the discovered process models.

Algorithms	Fitness	Precision	Size	CFC
Inductive Miner Infrequent (IMF)	0.98	0.60	49	46
Split Miner (SM)	0.98	0.80	**38**	41
Ours	**0.99**	**0.81**	41	**36**

Fig. 6 shows the process model discovered by the SM. Although the process model can still describe the ICU treatment process according to the conformance checking results, it is hard to recognise the loop structure between *ICU Admission* and *ICU Out*. For instance, domain experts cannot tell which is the "redo" part. The discovered process model is unstructured and hard to understand compared to our model, because it barely produces valuable insights for domain experts. The process model discovered by the IMF is presented in Fig. 7. According to Table.1, the model has lower precision and higher complexity. Unlike our method, the discovered loop structure misses *Invasive Monitoring*, which is discovered as a parallel activity with the loop structure. Hence, the model cannot accurately represent the process since *Invasive Monitoring* cannot happen at an arbitrary time during the routine ICU treatment process. Besides, none of the break behaviours are discovered. To summarise, existing process discovery methods have difficulty discovering such break behaviours. The discovered models are unstructured, hard to understand and possess relatively low precision and high complexity.

Goal 3: Further Analysis of ICU Discharge Behaviours More than 66.7 % patients are discharged within 48 hours, and no deaths are found among them. Most patients (64.4 %) are normally discharged from ICU after invasive

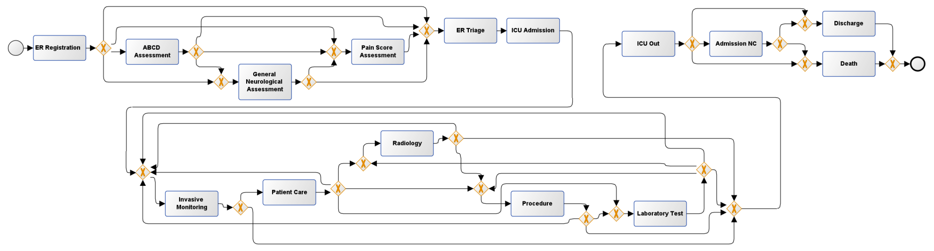


Fig. 6. The ICU pathways discovered by SM.

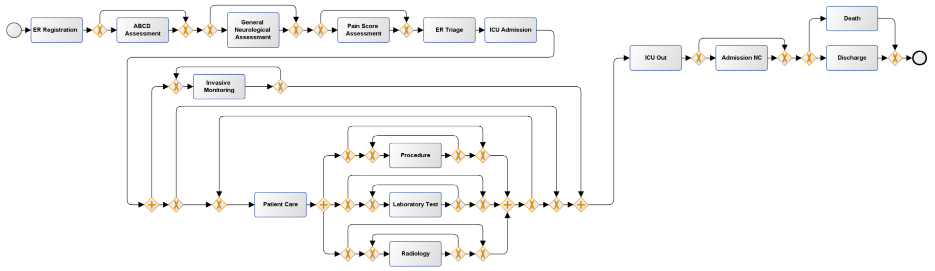


Fig. 7. The process model discovered by the IMF.

monitoring if their vital measurements are within the normal range. Some are further discharged from the hospital (i.e., transferred to other care facilities), and the remaining are admitted to normal wards in the hospital (i.e., *Admission NC*). Furthermore, three break discharge behaviours are discovered. 33.7 % of patients are discharged from ICU after performing further examinations ordered by doctors (i.e., laboratory tests and radiology), which is in line with the guideline [18]. Conversely, only 1.9 % of patients are discharged after specific procedures are conducted. Further investigation indicates that the most are discharged because of death. Regarding discharge time, we find that 74.2 % of patients are discharged in the morning, which indicates that although ICU usually have discharge rounds twice a day, the primary discharge decisions are made in the morning.

5 Discussion and Conclusion

This paper proposes a method to extend the IM framework to represent and discover break behaviours. The method is then applied to a healthcare event log to discover ICU discharge behaviours. Our method can discover more structured, understandable and accurate process models than existing process discovery algorithms.

It has to be noted that although our method can discover break behaviours in process trees, it may not always be possible to convert the break process trees into equivalent BPMNs/Petri nets. For example, in Fig.5, there will be remaining tokens in the loop after break behaviours. Future work is needed to represent the break behaviours in other process modeling notations (e.g., using cancellation regions to represent break behaviours). Finally, our method can be potentially applied to other domains. Further evaluation is needed to demonstrate the performance of our method.

References

1. van der Aalst, W., Weijters, T., Maruster, L.: Workflow mining: discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering **16**(9), 1128–1142 (Sep 2004)
2. Abo-Hamad, W.: Patient pathways discovery and analysis using process mining techniques: An emergency department case study. In: International conference on health care systems engineering. pp. 209–219. Springer (2017)
3. Augusto, A., Conforti, R., Dumas, M., La Rosa, M., Maggi, F.M., Marrella, A., Mecella, M., Soo, A.: Automated discovery of process models from event logs: review and benchmark. IEEE transactions on knowledge and data engineering **31**(4), 686–705 (2018)
4. Augusto, A., Conforti, R., Dumas, M., La Rosa, M., Polyvyanyy, A.: Split miner: automated discovery of accurate and simple business process models from event logs. Knowledge and Information Systems **59**(2), 251–284 (May 2019)
5. Benevento, E., Aloini, D., van der Aalst, W.M.: How can interactive process discovery address data quality issues in real business settings? evidence from a case study in healthcare. Journal of Biomedical Informatics **130**, 104083 (2022)
6. Berger, M.M., Reintam-Blaser, A., Calder, P.C., Casaer, M., Hiesmayr, M.J., Mayer, K., Montejo, J.C., Pichard, C., Preiser, J.C., van Zanten, A.R., et al.: Monitoring nutrition in the icu. Clinical nutrition **38**(2), 584–593 (2019)
7. Chen, Q., Lu, Y., Tam, C., Poon, S.: Predictive process monitoring for early predictions of short-and long-term mortality for patients with acute coronary syndrome. In: Pacific Asia Conference on Information Systems (2022)
8. Delias, P., Manolitzas, P., Grigoroudis, E., Matsatsinis, N.: Applying process mining to the emergency department. In: Encyclopedia of business analytics and optimization, pp. 168–178. IGI Global (2014)
9. Durojaiye, A.B., McGeorge, N.M., Puett, L.L., Stewart, D., Fackler, J.C., Hoonakker, P.L., Lehmann, H.P., Gurses, A.P.: Mapping the flow of pediatric trauma patients using process mining. Applied clinical informatics **9**(03), 654–666 (2018)
10. Ezzie, M.E., Aberegg, S.K., O'Brien Jr, J.M.: Laboratory testing in the intensive care unit. Critical care clinics **23**(3), 435–465 (2007)
11. Leemans, M., van der Aalst, W.M.P.: Modeling and Discovering Cancelation Behavior. In: Panetto, H., Debruyne, C., Gaaloul, W., Papazoglou, M., Paschke, A., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems. OTM 2017 Conferences. pp. 93–113. Lecture Notes in Computer Science, Springer International Publishing, Cham (2017)
12. Leemans, M., van der Aalst, W.M.P., van den Brand, M.G.J.: Recursion aware modeling and discovery for hierarchical software event log analysis. In: 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER). pp. 185–196 (Mar 2018)

13. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering Block-Structured Process Models from Incomplete Event Logs. In: Ciardo, G., Kindler, E. (eds.) Application and Theory of Petri Nets and Concurrency. pp. 91–110. Lecture Notes in Computer Science, Springer International Publishing, Cham (2014)
14. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Using Life Cycle Information in Process Discovery. In: Reichert, M., Reijers, H.A. (eds.) Business Process Management Workshops. pp. 204–217. Lecture Notes in Business Information Processing, Springer International Publishing, Cham (2016)
15. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Scalable process discovery and conformance checking. Software & Systems Modeling **17**(2), 599–631 (May 2018)
16. Leemans, S.J., Fahland, D., Van Der Aalst, W.M.: Discovering block-structured process models from event logs-a constructive approach. In: International conference on applications and theory of Petri nets and concurrency. pp. 311–329. Springer (2013)
17. Leemans, S.J., Fahland, D., Van Der Aalst, W.M.: Discovering block-structured process models from event logs containing infrequent behaviour. In: International conference on business process management. pp. 66–78. Springer (2013)
18. Lin, F., Chaboyer, W., Wallis, M.: A literature review of organisational, individual and teamwork factors contributing to the icu discharge process. Australian Critical Care **22**(1), 29–43 (2009)
19. Lohan, R.: Imaging of icu patients. In: Thoracic Imaging, pp. 173–194. Springer (2019)
20. Lu, Y., Chen, Q., Poon, S.: A Novel Approach to Discover Switch Behaviours in Process Mining. In: Leemans, S., Leopold, H. (eds.) Process Mining Workshops, vol. 406, pp. 57–68. Springer International Publishing, Cham (2021)
21. Lull, J.J., Cid-Menéndez, A., Ibanez-Sanchez, G., Sanchez, P.L., Bayo-Monton, J.L., Traver, V., Fernandez-Llatas, C.: Interactive process mining applied in a cardiology outpatient department. In: International Conference on Process Mining. pp. 340–351. Springer (2022)
22. Mans, R., Schonenberg, M., Song, M., Van der Aalst, W., Bakker, P.: Process mining in health care. In: International Conference on Health Informatics (HEALTHINF'08). pp. 118–125 (2008)
23. Munoz-Gama, J., Martin, N., Fernandez-Llatas, C., Johnson, O.A., Sepúlveda, M., Helm, E., Galvez-Yanjari, V., Rojas, E., Martinez-Millana, A., Aloini, D., et al.: Process mining for healthcare: Characteristics and challenges. Journal of Biomedical Informatics **127**, 103994 (2022)
24. Tam, C.S., Gullick, J., Saavedra, A., Vernon, S.T., Figtree, G.A., Chow, C.K., Cretikos, M., Morris, R.W., William, M., Morris, J., et al.: Combining structured and unstructured data in emrs to create clinically-defined emr-derived cohorts. BMC medical informatics and decision making **21**(1), 1–10 (2021)
25. Weijters, A., Ribeiro, J.: Flexible Heuristics Miner (FHM). In: 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM). pp. 310–317 (Apr 2011)
26. Zhou, Z., Wang, Y., Li, L.: Process mining based modeling and analysis of workflows in clinical care-a case study in a chicago outpatient clinic. In: Proceedings of the 11th IEEE international conference on networking, sensing and control. pp. 590–595. IEEE (2014)