Optimal Process Mining of Timed Event Logs

Hugo de Oliveira, Vincent Augusto, Baptiste Jouaneton, Ludovic Lamarsalle, Martin Prodel, Xiaolan Xie

Optimal Process Mining of Timed Event Logs

Hugo De Oliveira[a,b,*], Vincent Augusto[a], Baptiste Jouaneton[b], Ludovic Lamarsalle[b], Martin Prodel[b], Xiaolan Xie[a,c]

[a] *Mines Saint-Etienne, Univ Clermont Auvergne, CNRS, UMR 6158 LIMOS, Centre CIS, F - 42023 Saint-Etienne France*
[b] *HEVA, 186 avenue Thiers, F-69465, Lyon, France*
[c] *Antai College of Economics and Management, Shanghai Jiao Tong University, China*

Abstract

The problem of determining the optimal process model of an event log of traces of events with temporal information is presented. A formal description of the event log and relevant complexity measures are detailed. Then the process model and its replayability score that measures model fitness with respect to the event log are defined. Two process models are formulated, taking into account temporal information. The first, called grid process model, is reminiscent of Petri net unfolding and is a graph with multiple layers of labeled nodes and arcs connecting lower to upper layer nodes. Our second model is an extension of the first. Denoted the time grid process model, it associates a time interval to each arc. Subsequently, a Tabu search algorithm is constructed to determine the optimal process model that maximizes the replayability score subject to the constraints of the maximal number of nodes and arcs. Numerical experiments are conducted to assess the performance of the proposed Tabu search algorithm. Lastly, a healthcare case study was conducted to demonstrate the applicability of our approach for clinical pathway modeling. Special attention was paid on readability, so that final users could beneficially use the process mining results.

Keywords: process mining, event log, time modeling, tabu search, healthcare data, patient pathways.

1. Introduction

The digital revolution affects all industries and businesses and produces a huge amount of data. Numerous decision aid analytical methods and tools are available to take advantage on relevant data. Machine Learning methods have been widely used. Supervised and unsupervised learning for knowledge discovery, when applied to matrix data where each row is an observation characterized by features in columns, have spread over.

Knowledge discovery from healthcare data, such as patient lifetime hospitalization history is presently not optimal yet crucial. Therefore, we want to find common patterns, process models, or care pathways of hospitalization histories for cohorts honed to a specific time periods. Such studies are important to detect relevant "causal" relationships or transitions, to check the conformance of practice to guidelines, etc. An example of a causal relation could be "most patients of a given surgery had the same prior underlining condition or past medical event".

Traditional machine learning techniques are not well suited to generate process models from data. Process mining well suited for this purpose, and has been first formalized in 2004 [14], followed by developments in various fields [7] including healthcare [1]. Temporal information such as the time between two events and the number of repetitions of a given event in the past are particularly important in process modeling and prediction. In healthcare, a second hospital visit shortly after the first, unfortunately is likely to be an undesirable complication or result of the earlier admission. A patient having been hospitalized several times for a given disease or had a much longer prior hospital stay before recovery is more likely to need a

*Corresponding author
Email address: hdeoliveira@hevaweb.com (Hugo De Oliveira)

non-standard surgery than a patient having no or just one past hospitalization. Unfortunately, such features are rarely taken into account by process mining approaches. For example, repeated events are prohibited for the sake of visibility and time is not considered during model construction. A previous study on the understanding of process models found that the average connector degree and density are two identified factors which induce negative effects on comprehension (at a fixed size) [11].

Starting from these limitations regarding time consideration and comprehension, an extension of existing optimal process mining theory is presented. Thus, the main scientific contribution of this paper is the mathematical formalization of a new ascending and time-dependent process mining approach, structured on a grid for a better representation and understanding. Reminiscent of Petri net unfolding [2], ascending representation forbids loops and confusing backward transitions, whereas the time-dependent feature considers time patterns of event logs during optimization. Descriptors are introduced, characterizing event logs complexity and process models structure. The replayability indicator [10] is slightly modified and acts as a key performance indicator to evaluate the resulting process models. Finally, a new Tabu search procedure for process mining optimization is presented, tested and validated through a series of designed experiments as well as a real-life healthcare case study.

The rest of this paper is organized as follows. A literature review focused on recent work in process mining with a particular focus on healthcare is presented in Section 2. General definitions of event log representation are given in Section 3. A mathematical formulation of the new process models and optimization problems is presented in Section 4. Section 5 describes the process discovery methods involved and Section 6 details in depth computational experiments designed to test methods on different simulated event logs. A case study based on real data is presented in Section 7. Finally, conclusion and perspectives are given in Section 8.

2. Literature Review

The raison d'être of Process mining is to discover, monitor and ameliorate actual processes as they occur by extracting knowledge from event logs readily available in I.T. systems. Process mining is situated between Big Data and Data Mining on one side, and Business Process Modeling and Analysis on the other. The field of process mining can be divided in 3 main areas: process discovery, conformance checking and extension of a model [13].

Recently, two systematic studies mapped out Process Mining [7, 1], which are valuable to describe the scope and the dynamics of this field. Maita et al. regrouped in their study 705 papers about process mining from 2005 to 2014 [7]. The number of publications addressing process mining has significantly increased from 2005 to 2014. "Discovery" is the main purpose for the use of process mining (71% of papers), with "graph structure-based techniques" being the most common intersection. For studies mentioning a specific application domain, "Medicine and Healthcare" is the second most frequent domain just after the overall sector "Enterprise". Moreover, "clinical analysis pathway" ranks third as data sources used for case studies or experiments. Importantly, the majority of these studies does not mention any process mining tool being used. For the rest, `ProM` is the most often mentioned with in-house frameworks a distant second.

A similar work [1] on healthcare studies considers 172 papers from 2005 to 2017. Observations detail a rapid expansion of the field, giving new opportunities for further research and practice. Process discovery appears as the most important activity of process mining when applied to healthcare. Studies on "Healthcare process" (93) are more common than "Clinical pathway" (59). Furthermore, studies corresponding to "Multiple hospital" data are less frequent (14) compared to "Single hospital" data (130). The same observation is true for studies regarding "Multiple department" data (13) and "Single department" data (122). Finally, the number of studies which propose a new tool, model or metric is low (17%). This limited body of relevant papers in our field seems to imply the originality of our contribution.

Since the later systematic review on healthcare, new process mining contributions applied to healthcare have been proposed. Kusuma et al. compiled a literature review of process mining in cardiology [5]. Promising opportunities to assist medical experts in care analysis were shown, although few formal process mining methodologies were included. Litchfield et al. [6] introduced a study protocol to apply process mining to primary care in the UK for the first time. The use of orthodox process mapping in addition to data-driven process mining is presented as useful to identify differences and similarities.

Similar works on patient pathways mining were published using discrete optimization [8] and simulation [9]. The most related paper to our work is of Prodel et al. in 2018, where a mathematical formulation of the problem was presented, along with a Tabu search optimization process to search for best process model. In addition, to reduce the computation time for large-scale data sets, they used a Monte Carlo sampling method. As healthcare data often contains hierarchical structure (ICD-10 codes for example), this data characteristic for labels was considered during the optimization process [9]. The large scale problematic and the hierarchical structure of labels were also addressed.

There are two limitations. First, qualitative representation of pathways with repeated events is not readily understood by decision makers and clinicians. Indeed, a repeated event pattern in the event log will be represented by a loop, which does not take into account the linear characteristic of patient pathways. More importantly, the time is not considered during the optimization process. The including of time as a parameter could be beneficial to highlight hidden time patterns contained in the event log.

Therefore, two extended process mining approaches are presented in this paper. New descriptors to characterize data sets and newly formulated process models are also presented. The Tabu search used by Prodel et al. [10] is tested and compared to other methods. Moreover, an improved version of the Tabu Search algorithm adapted to new process model is suggested.

3. Event, Trace and Log

This section provides a formal description of data involved including events, traces and event logs.

Definition 1. *(Event). Each event denoted e is a couple (a, t) where $a \in A$ is an element of a finite set A of labels corresponding to the event class of e, and $t \in T$ with $T = \mathbb{N}$ or $\mathbb{R}$ corresponding to the event time also called time-stamp. An event e is then equivalently defined by the two following functions:*

- $label(e) = a$ *called labeling function;*
- $time(e) = t$ *called timing function.*

Definition 2. *(Trace). A trace is a sequence of events denoted as $\sigma = e_1, e_2, \ldots, e_m$ with $m \in \mathbb{N}^*$ such that $e_k \in A \times T$ and $time(e_k) < time(e_{k+1})$. A trace leads to the following functions:*

- $trace(e_k)$ *denoting the trace ID of each event;*
- $position(e_k) = k$ *denoting the order of the event in the trace;*
- $|\sigma| = m$ *denoting the length of the trace.*

Definition 3. *(Event log). An event log is a set of traces denoted as $L = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ with $n \in \mathbb{N}^*$. An event log contains all input data of this paper. Without loss of generality, we assume that each label appears at least once in the event log L, i.e. $\forall a \in A : \exists \sigma \in L, e \in \sigma \mid e = (a, t)$. The set of possible combinations of labels and positions in L is:*

$$A_{lab,pos} = \bigcup_{\sigma = e_1, \ldots, e_m \in L} \{(label(e_1), 1), \ldots, (label(e_m), m)\}$$

Definition 4. *(Causal relations). For each trace $\sigma = e_1, \ldots, e_m$ in L, all pairs of labels $(label(e_k), label(e_l))$ such that $1 \leq k < l \leq m$ are called causal relations and pairs $(label(e_k), label(e_{k+1}))$ are called direct causal relations. Let T^C be the set of all causal relations and T^{DC} be the set of direct causal relations.*

Definition 5. *(Timed causal relations). All triplets $(label(e_k), label(e_l), time(e_l) - time(e_k))$ such that $(label(e_k), label(e_l)) \in T^C$ (or T^{DC}) are called timed causal relations (or direct timed causal relations). Let T^{tC} and T^{tDC} be the set of all timed causal relations and the set of direct timed causal relations.*

Definition 6. *(Diversity measures). The event log L has the following diversity measures:*

- *event diversity* $div_e = |A|$;
- *event-position diversity* $div_{e,p} = |A_{lab,pos}|$;

Event log definitions
Label set: A
Event: $e_k = (a, t)$
$a \in A$, $t \in T$ with $T = \mathbb{N}$ or $\mathbb{R}$
Trace: $\sigma = e_1, ..., e_m$
Log: $L = \{\sigma_1, ..., \sigma_n\}$
Label: $label(e_k) = a$
Time: $time(e_k) = t$
ID: $trace(e_k)$
Position: $position(e_k) = k$
Relation sets
Causal relations set: $T^C = \{(label(e_k), label(e_l))\}_{\substack{1 \leq k < l \leq m \\ \sigma = e_1, ..., e_m \in L}}$
Direct causal relations set: $T^{DC} = \{(label(e_k), label(e_{k+1}))\}_{\substack{1 \leq k < m \\ \sigma = e_1, ..., e_m \in L}}$
Timed causal relations set: $T^{tC} = \{(label(e_k), label(e_l), time(e_l) - time(e_k))\}_{\substack{1 \leq k < l \leq m \\ \sigma = e_1, ..., e_m \in L}}$
Timed direct causal relations set: $T^{tDC} = \{(label(e_k), label(e_{k+1}), time(e_{k+1}) - time(e_k))\}_{\substack{1 \leq k < m \\ \sigma = e_1, ..., e_m \in L}}$
Descriptors
Traces length description:
$\lvert\sigma\rvert_{mean}$, $\lvert\sigma\rvert_{std}$, $\lvert\sigma\rvert_{max}$, $\lvert\sigma\rvert_{min}$
Event diversity: $div_e = \lvert A \rvert$
Event-position diversity: $div_{e,p} = \lvert A_{lab,pos} \rvert$
Causal relation diversity: $div_{causal} = \lvert T^{DC} \rvert$
Timed causal relation diversity: $div_{t-causal} = \lvert T^{tDC} \rvert$

Table 1: Event log notations.

- *causal relation diversity* $div_{causal} = |T^{DC}|$*;*
- *timed causal relation diversity* $div_{t-causal} = |T^{tDC}|$.

Proposition 1. If $A \neq \emptyset$ and $|\sigma| > 1$ for at least one trace σ, then $1 \leq div_e \leq div_{e,p} \leq \sum_{\sigma \in L} |\sigma|$ and $1 \leq div_{causal} \leq div_{t-causal} \leq \sum_{\sigma \in L} (|\sigma| - 1)$

Proof. Trivial by definitions. □

The diversity measures characterize the log's complexity: a high diversity means an increased number of different elements in terms of labels and causal relations. Also, the distribution of trace lengths is a valuable predictor to assess event logs' complexity (let $|\sigma|_{mean}$, $|\sigma|_{std}$, $|\sigma|_{max}$ and $|\sigma|_{min}$ be its mean, standard deviation, maximum and minimum values, respectively). Notations of event logs are summarized in Table 1.

Example 1. *Table 2 shows a short event log related to patient hospitalization pathways. Each row is a hospitalization event. Events with the same ID are ordered by increasing time stamp and represent a trace. Each trace is a patient's hospitalization history. Each first event of a patient has a timestamp set to* 0*. By clustering hospitalization by main diagnosis, the set of labels is as follows :*

- $A_{lab} = \{$*I500, I44.2, I621, G935, E149, I272*$\}$.

Alternative event clustering is possible by taking into account more detailed information such as the duration and secondary treatments during the hospitalization.

Thus, event-position set is as follows:

- $A_{lab,pos} = \{$*(I500, 1), (I44.2, 2), (I621, 3), (G935, 4), (E149, 2), (I272, 3), (I500, 4), (I44.2, 5)*$\}$.

ID	time-stamp	duration	main diagnosis	DRG
0	0	8	I500	05M092
0	30	0	I44.2	05C142
0	72	1	I621	01M311
0	103	3	G935	01M131
1	0	2	I500	05M092
1	5	0	E149	10M02T
1	93	2	I272	05M172
1	145	1	I500	05M092
1	180	8	I44.2	05C142

Table 2: An example of an event log of patient pathways.

Basic and timed transitions are:

- $T^C = \{$*(I500, I44.2), (I500, I621), (I500, G935), (I44.2, I621), (I44.2, G935), (I621, G935), (I500, E149), (I500, I272), (I500, I500), (E149, I272), (E149, I500), (E149, I44.2), (I272, I500), (I272, I44.2))*$\}$
- $T^{DC} = \{$*(I500, I44.2), (I44.2, I621), (I621, G935), (I500, E149), (E149, I272), (I272, I500))*$\}$;
- $T^{tC} = \{$*(I500, I44.2, 30), (I500, I621, 72), (I500, G935, 103), (I44.2, I621, 42), (I44.2, G935, 73), (I621, G935, 31), (I500, E149, 5), (I500, I272, 93), (I500, I500, 145), (I500, I44.2, 180), (E149, I272, 88), (E149, I500, 140), (E149, I44.2, 175), (I272, I500, 52), (I272, I44.2, 87), (I500, I44.2, 35))*$\}$;
- $T^{tDC} = \{$*(I500, I44.2, 30), (I44.2, I621, 42), (I621, G935, 31), (I500, E149, 5), (E149, I272, 88), (I272, I500, 52), (I500, I44.2, 35))*$\}$.

Finally, diversity descriptors are:

- $div_e = 6$; $div_{e,p} = 8$; $div_{causal} = 6$; $div_{t-causal} = 7$.

4. Grid Process Model Optimization Problem

This section is dedicated to discover process models that best fit the event logs given in Section 3. For this purpose, first the concepts of grid process models and time grid process models are introduced. Subsequently, the fitness measure is introduced to measure how well a process model captures the causal relations of an event log. We terminate by a formal definition of the process model optimization problem.

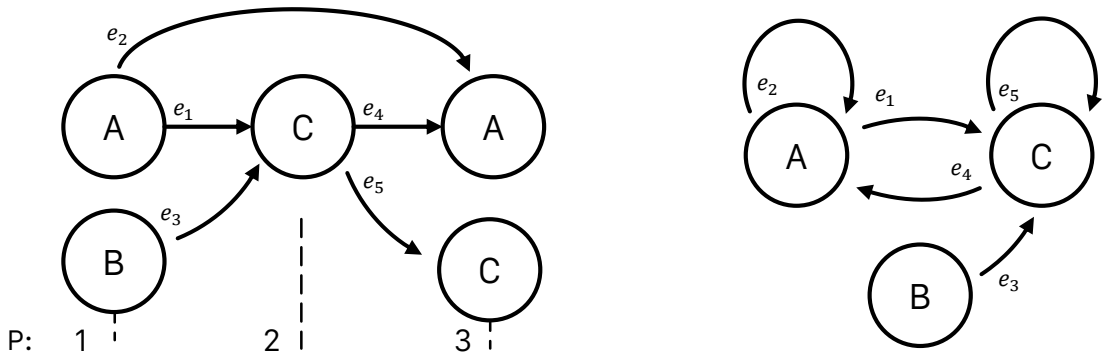
4.1. Grid Process Models

Definition 7. *(Grid process model). A grid process model of a given log L is a triplet $G\text{-}PsM = (N, E, \mathcal{L})$ where:*

- *N is a set of nodes partitioned into K disjoint subsets called layers, i.e. $N = N_1 \cup \cdots \cup N_k, N_k \cap N_l = \emptyset$;*
- *$E \subset N \times N$ is a set of edges such that $(x, y) \subset E$ with $x \in N_k, y \in N_l$ implies $k < l$, i.e. the process model is acyclic with edges going from lower layers to higher layers;*
- *$\mathcal{L} : N \to A$ is the labeling function of the nodes.*

From the above definition, one can also define the position function $\mathcal{P}(x) = k$, if $x \in N_k$. As a result, $(x, y) \subset E$ implies $\mathcal{P}(x) < \mathcal{P}(y)$.

The main difference from the previous process model definition in [10] is the possibility for an event to appear at various positions in a trace. For example, an event happening both at the beginning and at the end of a patient pathway could now be described in the process model by two nodes, one with low and the other with a high position. In this case, loops on the same node and backward edges are not allowed anymore. An example of a $G\text{-}PsM$ and its process model equivalent are given in Figure 1.



(a) Grid process model.

(b) Equivalent classic process model.

Figure 1: Example of a grid process model and its classic process model equivalent.

Example 2. *The Figure 1a shows a grid process model G-PsM* $= (N, E, \mathcal{L})$ *with:*

$$N = (n_1, n_2, n_3, n_4, n_5) \quad E = (e_1, e_2, e_3, e_4, e_5)$$

$$\begin{array}{lll} \mathcal{L}(n_1) = A & \mathcal{P}(n_1) = 1 & e_1 = (n_1, n_3) \\ \mathcal{L}(n_2) = B & \mathcal{P}(n_2) = 1 & e_2 = (n_1, n_4) \\ \mathcal{L}(n_3) = C & \mathcal{P}(n_3) = 2 & e_3 = (n_2, n_3) \\ \mathcal{L}(n_4) = A & \mathcal{P}(n_4) = 3 & e_4 = (n_3, n_4) \\ \mathcal{L}(n_5) = C & \mathcal{P}(n_5) = 3 & e_5 = (n_3, n_5) \end{array}$$

An equivalent classic process model is presented in Figure 1b, without duplicated label nodes, allowing loops and backward transitions.

4.2. Time Grid Process Models

To the best of our knowledge, in existing process mining approaches, time-related information is added after model discovery and remains descriptive. Significantly, we provide a new approach to include time-related information within the optimization process of building a process model.

Definition 8. *(Time grid process model). A time grid process model of a given log L is a four-uplet TG-PsM* $= (N, E, \mathcal{L}, \mathcal{T})$ *where:*

- $(N, E, \mathcal{L})$ *is a grid process model with eventually multiple edges between nodes;*
- $\mathcal{T} : E \to T \times T$ *associates a time interval* $[a_{(x,y)}, b_{(x,y)}]$ *to each edge* $(x, y) \in E$.

As shown in Figure 2, using this definition, previous unique edges between two nodes in *G-PsM* are replaced by multiple possible edges in *TG-PsM*, each of them having its own time interval. Definition 8 ensures that a given causal relation, with a given time value between two events, will be characterized by a unique possible edge with same starting and ending node in the process model. The uniqueness of characterization will be useful for graph construction and replayability defined thereafter.

Example 3. *The Figure 2 shows a time grid process model TG-PsM* $= (N, E, \mathcal{L}, \mathcal{T})$ *with:*

$$N = (n_1, n_2, n_3, n_4) \quad E = (e_1, e_2, e_3, e_4, e_5, e_6, e_7)$$

$$\begin{array}{llll} \mathcal{L}(n_1) = A & \mathcal{P}(n_1) = 1 & e_1 = (n_1, n_2) & \mathcal{T}(e_1) = [10, 24] \\ \mathcal{L}(n_2) = B & \mathcal{P}(n_2) = 2 & e_2 = (n_1, n_2) & \mathcal{T}(e_2) = [30, 35] \\ \mathcal{L}(n_3) = C & \mathcal{P}(n_3) = 2 & e_3 = (n_1, n_2) & \mathcal{T}(e_3) = [45, 62] \\ \mathcal{L}(n_4) = A & \mathcal{P}(n_4) = 3 & e_4 = (n_2, n_4) & \mathcal{T}(e_4) = [0, 5] \\ & & e_5 = (n_2, n_4) & \mathcal{T}(e_5) = [5, 25] \\ & & e_6 = (n_3, n_4) & \mathcal{T}(e_6) = [2, 50] \\ & & e_7 = (n_3, n_4) & \mathcal{T}(e_7) = [75, 100] \end{array}$$

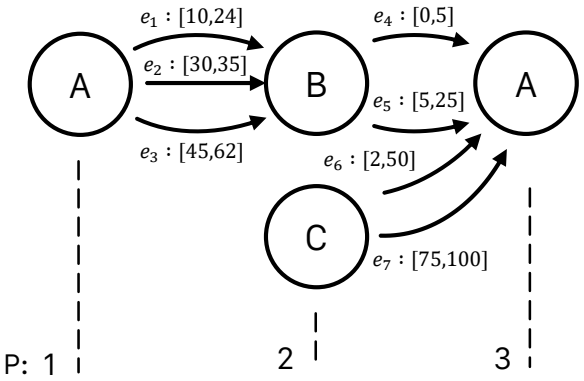


Figure 2: Example of a time grid process model.

G-PsM	**TG-PsM**
$G\text{-}PsM = (N, E, \mathcal{L})$	$TG\text{-}PsM = (N, E, \mathcal{L}, \mathcal{T})$
$N = N_1 \cup \cdots \cup N_k$	$N = N_1 \cup \cdots \cup N_k$
$\mathcal{L}(x) \in A, x \in N$	$\mathcal{L}(x) \in A, x \in N$
$\mathcal{P}(x) = k, x \in N_k$	$\mathcal{P}(x) = l, x \in N_l$
	$\mathcal{T}((x,y)) = [a_{(x,y)}, b_{(x,y)}], (x,y) \subset E$
$Div_N = \lvert\{\mathcal{L}(n)\}_{n \in N}\rvert$	$Div_N = \lvert\{\mathcal{L}(n)\}_{n \in N}\rvert$
	$Div_E = \lvert\{(\mathcal{L}(x), \mathcal{P}(x),$
	$\mathcal{L}(y), \mathcal{P}(y))\}_{(x,y) \subset E}\rvert$

Table 3: Process model related notations.

4.3. Process Model Complexity Characterization

The larger a process model is, the better it represents traces from an event log, but the drawback is that the more convoluted it becomes for someone to understand. Building a process model while controlling its complexity is crucial. A process model's complexity is described by its number of nodes $|N|$ and edges $|E|$. Process model diversities are defined similarly to event log diversities (Definition 6).

Definition 9. *(Node and edge diversities). For a process model G-PsM or TG-PsM, the node diversity is* $Div_N = |\{\mathcal{L}(n)\}_{n \in N}|$. *For a time grid process model TG-PsM, the edge diversity is* $Div_E = |\{(\mathcal{L}(x), \mathcal{P}(x), \mathcal{L}(y), \mathcal{P}(y))\}_{(x,y) \subset E}|$.

Diversity descriptors characterize the variety of nodes and edges of a process model. A high diversity means that only few nodes (or edges) have the same label, whereas a low diversity indicates many similarly labeled nodes (or edges). For instance, in the $G\text{-}PsM$ of Figure 1a, $|N| = 5$, $Div_N = 3$ and $|E| = 5$, and in the $TG\text{-}PsM$ presented in Figure 2, $|N| = 4$, $Div_N = 3$, $|E| = 7$ and $Div_E = 3$. This simple example highlights the increase of $|E|$ for $TG\text{-}PsM$ compared with that of $G\text{-}PsM$. Notations of the process models are summarized in Table 3.

4.4. Replayability

To evaluate the capacity of a process model to represent a trace, a new replayability score has been devised to match the newly defined grid process models. Initially, preliminary definitions are required.

Definition 10. *(Replayability). An event e is said replayed by a process model G-PsM if* $label(e) = \mathcal{L}(x)$ *for some node x of G-PsM. A causal relation* (e_k, e_l) *is said basic replayed by G-PsM if* $label(e_k) = \mathcal{L}(x)$ *and* $label(e_l) = \mathcal{L}(y)$ *for some edge* (x, y) *of G-PsM. A timed causal relation* $(e_k, e_l, time(e_l) - time(e_k))$ *is said time-replayed by an TG-PsM if* $label(e_k) = \mathcal{L}(x)$, $label(e_l) = \mathcal{L}(y)$ *and* $time(e_l) - time(e_k) \in [a_{(x,y)}, b_{(x,y)}]$ *for some edge* (x, y) *of the TG-PsM.*

Algorithm 1 Grid and Time grid replayability games.

1: **Initialization**
2: $z \leftarrow 0$, $\delta \leftarrow 0$, $\phi \leftarrow 0$, $m \leftarrow 0$
3: Find c_m the first replayed event of σ
4: If no event of σ replayed:
5: **return** z, δ, ϕ
6: Else:
7: Set N_{actual} as the node which replayed c_m
8: $z \leftarrow z + 1$
9: **Trace crossing**
10: While $m < |\sigma|$:
11: If c_{m+1} can be replayed in *G-PsM* by a node N_{next} with $\mathcal{P}(N_{actual}) < \mathcal{P}(N_{next})$:
12: $z \leftarrow z + 1$, $N_{actual} \leftarrow N_{next}$ (with the lowest position)
13: If no edge exists between N_{actual} and N_{next} (or if the transition between (c_m, c_{m+1}) cannot be time replayed *(Time grid replayability game)*):
14: $\phi \leftarrow \phi + 1$
15: $m \leftarrow m + 1$
16: **Skipped elements analysis**
17: If at least one element of σ has been skipped (not been replayed between replayed elements):
18: $\delta = 1$
19: **Conclusion**
20: **return** z, δ, ϕ

Remark 1. For any transition in a trace we have, by definition, {time replayability} $\Rightarrow$ {basic replayability} but {basic replayability} $\not\Rightarrow$ {time replayability}.

To calculate the replayability of a trace, an algorithmic procedure is presented, named **Grid replayability game** (or **Time grid replayability game**) (Algorithm 1). The grid replayability game starts from c_m, with m being the index of the first event of σ replayed in *G-PsM* (line 3). If c_m is replayed by several nodes of *G-PsM*, the node with the lowest position is chosen (line 7). The next event c_{m+1} of σ possibly replayed by *G-PsM* is sought, with a strictly superior node position than the previous node which replayed c_m (lines 11–12). If the transition (c_m, c_{m+1}) is not basic-replayed by *G-PsM*, the transition is said **strongly-forced**. If the transition $((c_m, c_{m+1}), t_{m,m+1})$ is not time-replayed by *TG-PsM*, the transition is said **time-forced** (lines 13–14). This process is repeated until the last replayable event is reached (line 10). If at least one event of σ has not been replayed while being in between two replayed events, it is said **skipped** (lines 17–18).

The strictly ascending condition of Definition 7 ensures that transitions are achieved by increasing positions during the replayability game. Thus, some events of σ might not be replayed during the game, even if they would have been according to Definition 10. Based on these new replayability games, adapted replayability score functions are introduced.

Definition 11. *(Replayability score). Considering the Grid replayability game, the replayability score of a sequence σ in a G-PsM or TG-PsM is defined as follows:*

$$R(\text{G-PsM}, \sigma) \text{ or } R(\text{TG-PsM}, \sigma) = (\frac{z}{|\sigma|} - \alpha * \delta - \beta * \frac{\phi}{|\sigma|})^+$$

where:

- *z is the number of events of σ replayed by G-PsM;*
- *δ a binary variable equal to 0 if no event of σ is skipped;*
- *ϕ is the number of (timed) strongly-forced transitions;*
- *α, β are weighting factors.*

Proposition 2. For a given trace σ, a process model *TG-PsM* and a *G-PsM* obtained with nodes and simple edges of *TG-PsM*:

$$R(\text{TG-PsM}, \sigma) \leq R(\text{G-PsM}, \sigma) \tag{1}$$

Proof. Proposition 2 translates the strictest character of the time grid replayability compared to grid replayability, for fixed coefficients. □

4.5. Problem Formulation for Process Model Discovery

Let L be an event log, the process model optimization problem consists in determining an optimal grid process model $G\text{-}PsM$ defined on L, maximizing the replayability and under some process model complexity constraints.

$$(\text{GridOpt}) \max_{\text{G-PsM}=(N,E,\mathcal{L})} R(\text{G-PsM}, L) \tag{2}$$

with $R(\text{G-PsM}, L) = \frac{1}{|L|} \sum_{\sigma \in L} R(\text{G-PsM}, \sigma)$

subject to

$$E \subseteq N \times N \tag{3}$$

$$max(\mathcal{P}(x)_{x \in N}) \leq p_{max} \tag{4}$$

$$|N| \leq U_N \tag{5}$$

$$|E| \leq U_E \tag{6}$$

where $R(\text{G-PsM}, \sigma), R(\text{G-PsM}, L) \in [0, 1]$, $p_{max} \in \mathbb{N}^*$ is the maximum position for the process model construction, $U_N \in \mathbb{N}^*$ and $U_E \in \mathbb{N}$ are the process model node and edge complexity bounds, respectively.

The problem of determining an optimal time grid process model, denoted as *TimeGridOpt* is similar, with the following supplementary constraint:

$$\forall e \in E, \mathcal{T}(e) \in T_E(e) \tag{7}$$

where $T_E(e) = ([a_j, b_j])_{j \in \mathbb{N}^*}$ is a pre-defined set of disjointed intervals, with $\forall e \in E$, $\forall I = [a, b] \in T_E(e)$, $a, b \in \mathbb{N}$ and $a < b$.

The purpose of this constraint is to have accurate but specific values of time intervals for each edge. Thus, we first define judicious intervals for each edge, before optimization of the time grid replayability score. For example, let x and y be two nodes of a time grid process model $TG\text{-}PsM$. After looking at all possible causal transitions $(\mathcal{L}(x), \mathcal{L}(y))$ in event log L, we obtain $\mathcal{D}$ the distribution of corresponding time values. Let m^- and m^+ be the maximum and minimum values of $\mathcal{D}$, respectively. An optimal solution in terms of complexity and replayability is to take the single edge with the full interval $[m^-, m^+]$. Inconveniently, this does not highlight time particularities and specific intervals. This is why dividing the interval $[m^-, m^+]$ in relevant sub-intervals $\{[x_i, y_i]\}_i$ where $\min_i x_i \geq m^-$, $\max_i y_i \leq m^+$ and $\bigcap_i \{[x_i, y_i]\} = \emptyset$ would prove more advantageous.

For both problems, an optimal process model has to be found in terms of nodes (with their labels and positions) and (time-) edges (basic or with a specific time interval). Complexity being a hard constraint, it can be useful to first define an optimal solution without complexity constraints, to illustrate the complexity of a potential optimal model.

Proposition 3. For any log L with $p_{max} = max_{\sigma \in L}|\sigma|$, $U_N = div_{e,p}$ and $U_E = \sum_{\sigma \in L}(|\sigma| - 1)$, there exists a process model $G\text{-}PsM$ such that $R(G\text{-}PsM, L) = 1$.

Proof. All traces are perfectly replayed in a process model with $A_{lab,pos}$ as the set of nodes and arcs connecting nodes of position p to position $p+1$ corresponding to direct causal relations from events of position p in some trace. □

The Figure 3 shows an example of the process models described previously, with $A_{lab} = \{A, B, C, D, E, F\}$ and $|\sigma_{max}| = 6$.

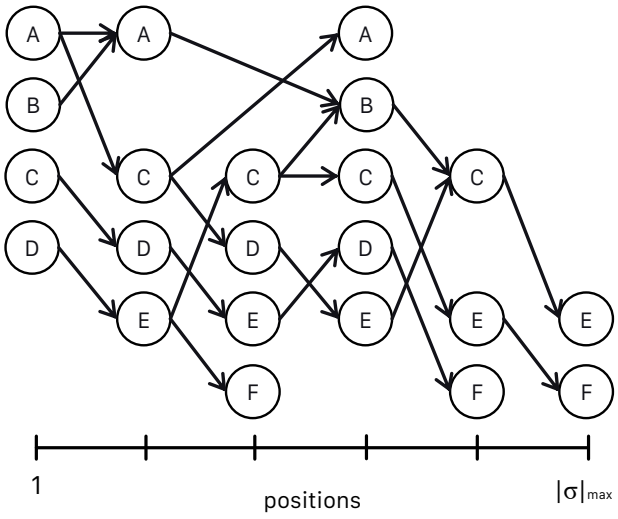


Figure 3: Example of a process model described in Proposition 3.

4.6. Property of the Replayability Game

Proposition 4. *(Optimized edges configuration property)* For any given process model (*G-PsM* or *TG-PsM*) and for any trace σ, the nodes reached during the replayability game are independent of the edges of the process model.

Proof. Trivial as, at each step of the replayability game, the closest event of σ with label equal to the label of a node of higher position gives the next node. This is independent of arcs. □

5. Optimization and Process Discovery

This section presents five algorithms : Random Search (RS), Frequent Model (FM), Spring Search (SS), Tabu Search (TS) and Tabu Search with Optimal Edges (TSOE). These algorithms are used to solve the optimization problems *GridOpt* and *TimeGridOpt* (defined through Equation 2). Among them, one heuristic (SS) and two meta-heuristics (TS and TSOE) are tested, modifying a current graph solution (or creating a neighborhood of new solutions). Transformation of the current solution is done by achieving some moves of nodes and edges. Before presenting these methods, the data preparation process is detailed in the following.

5.1. Data Preparation

To select nodes or edges to add (or delete) during the optimization process, a function $f_n : A \times [\![1, p_{max}]\!] \to \mathbb{N}$ is defined. For each tuple (l, p) with $l \in A$ and $p \in [\![1, p_{max}]\!]$, a value corresponding to its number of appearances within the event log is assigned. An event's position is either its real position in its trace if $|\sigma|_{max} \leq p_{max}$, or a rescaled value to ensure all positions to be between 1 and p_{max} if $|\sigma|_{max} > p_{max}$. Similarly, a function $f_e : T^C \to \mathbb{N}$ is defined regarding appearance of transitions in event log. An extension for *TG-PsM* is also proposed, considering transitions and time intervals to compute f_e values. In the following, "*promising*" is employed to characterize a node $n = (l, p)$ or an edge $e = (x, y)$ (with $\mathcal{T}(e)$ for *TG-PsM*) with a high value for $f_n(l, p)$ or $f_e(\mathcal{L}(x), \mathcal{L}(y))$. These functions are useful to select *promising* nodes or edges to add during the optimization process.

Algorithm 2 Tabu search with Optimized Edges (TSOE) for *GridOpt*.

Step 1 – Initialization
  Select an initial random unconnected solution: s_0^*
  Create the best connected solution s_0 from s_0^*
    Compute Grid replayability game for each patient
    Identify among possible edges the most frequently used
    Add these edges to the final model: s_0
  Compute replayability for the initial solution: $R(s_0, L)$
  Update the best known solution: $s_{best} \leftarrow s_0$
  Create initial Tabu list of unconnected solutions : $TL = [s_0^*]$
Step 2 – Iteration
  Generate from current unconnected solution s^* a neighborhood of unconnected non-tabu solutions : $N_H^*(s^*)$
  Generate best connected solutions from $N_H^*(s^*)$: $N_H(s^*)$
  Compute replayability of each element of $N_H(s^*)$
  Select the new solution as the neighbor with the highest value of replayability: s_{new}
  Update current solution : $s \leftarrow s_{new}$
  Update Tabu list: $TL \leftarrow TL + [s_{new}^*]$
    If $R(s_{best}, L) < R(s, L)$:
      $s_{best} \leftarrow s$
Step 3 – Repeat step 2 until a stopping criterion is reached

5.2. Spring Search (SS) for GridOpt

A simple greedy heuristic, called Spring search (iterative jumps) is initially utilized to solve the *GridOpt* problem. During the search, new solutions are proposed by iteratively increasing and decreasing the size of a solution, which brings diversity Each iteration consists of four steps: (1) delete all edges of the current solution, (2) delete K *non-promising* nodes, (3) add K new *promising* nodes, and (4) add U_E edges (the edge complexity bound), selected from the subset of possible edges (depending on nodes present in the $G\text{-}PsM$). At the end of an iteration, the obtained $G\text{-}PsM$ becomes the current solution.

5.3. Tabu search (TS) for GridOpt

Similarly to [10], a Tabu search is implemented to solve *GridOpt*. Because the strictly ascending condition increases the dependence of edges towards nodes, only one move is used to generate neighborhoods. At each iteration, a neighborhood is made of non-tabu neighbors which are obtained from the current solution in 4 steps : (1) delete a *non-promising* node and its surrounding edges, (2) consider the number of deleted edges as a budget X to reassign, (3) add a *promising* new node, and (4) add X new *promising* edges respecting the strictly ascending condition. Each neighbor is evaluated by computing replayability, and the best neighbor is kept as current solution.

5.4. Tabu Search with Optimized Edges (TSOE) for GridOpt

According to Proposition 4, edges do not intervene in the choice of the next node reached during the replayability game. Considering a solution without edges $G\text{-}PsM^*$, the replayability $R(G\text{-}PsM^*, \sigma)$ of a trace $\sigma \in L$ will have all possible transitions from a node x to a node y forced during the replayability game. If an edge from x to y is added to the solution, the replayability score $R(G\text{-}PsM^*, L) = \frac{1}{|L|}\sum_{\sigma \in L} R(G\text{-}PsM^*, \sigma)$ will increase, by a coefficient $c_{(x,y)} = \frac{\beta}{|L|}\sum_{\sigma \in L} \frac{f_\phi(\sigma,(x,y))}{|\sigma|}$, with $f_\phi(\sigma, (x, y)) = 1$ if transition (x, y) has been forced during the replayability game of σ, 0 otherwise. Thus, by computing $c_{(x,y)}$ for every possible transition (x, y), keeping top-U_E transitions produces the best-edge configuration from $G\text{-}PsM^*$, respecting constraint (6) of Equation 2.

From these observations, an adapted version of Tabu Search is defined, consisting in searching only for solutions without edges, and then for every solution in evaluating the best edges to add for an optimized replayability score. This method's advantage drastically reduces the search space to graphs without edges. Tabu Search with Optimized Edges is further detailed in Algorithm 2.

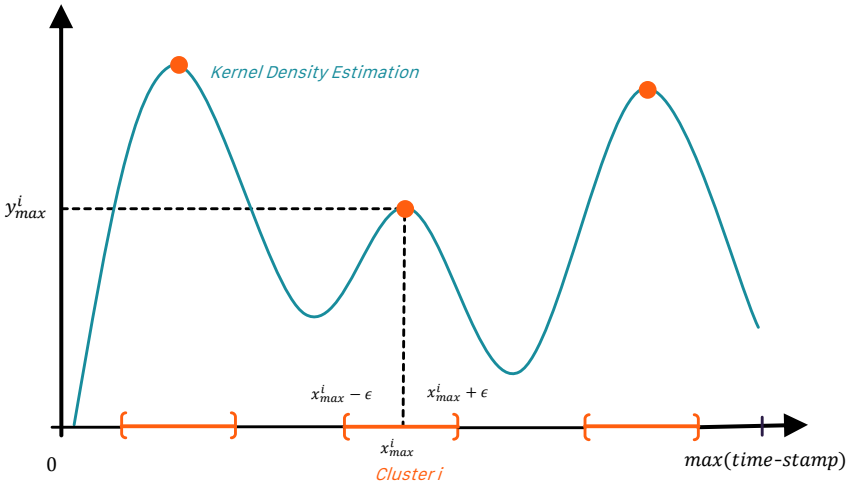


Figure 4: Illustration of KDE clustering.

5.5. TSOE for TimeGridOpt

Before solving *TimeGridOpt*, accurate but specific values of time intervals $T_E(e)$ for possible edges $e \in E$ need to be defined. First, causal transitions of event logs are analyzed to determine corresponding time distributions. Thus, a 1-D clustering method based on Kernel Density Estimation (KDE) is used to construct the set of possible edges to add during the process model optimization. KDE is the construction of an estimate of the density function from observed data, using a kernel function [12]. Applied to each time distribution, the resulting function will be used to define clusters by considering local maxima of the function as centers of clusters. For each maximum, an interval $[x_{max} - \epsilon, x_{max} + \epsilon]$ is defined, as shown in Figure 4. A low value of ϵ gives small and precise clusters; a high value decreases precision, replays more transitions and increases the global replayability score.

Finally, the approach is the same as for TSOE *GridOpt*, except that edges to add come with the most suited time interval as found by 1-D clustering. For an unconnected solution, the algorithm will choose the top-U_E timed-edges to get the final process model solution.

6. Computational Experiments

6.1. Log Generation

The following design of experiments, to test and compare the performances of the previously defined algorithms on various event logs, is presented here. Event logs of various sizes were generated to match real-life variability in data sets. All experiments were performed on an Intel Core i7 processor (2.8 GHz), 16 GB RAM, and Windows 10 OS. The algorithms were coded in Python 3.6.

6.1.1. Log Generation for GridOpt (without time)

Event logs are generated from process models, which must be created first. Three parameters are required to create a *G-PsM*: (i) a maximum position p_{max}, (ii) an event diversity div_e, and (iii) an event-position diversity $div_{e,p}$. For a given combination of these three, a fully connected *G-PsM* is randomly created with all the edges respecting the strictly ascending condition. Then, traces are generated by selecting a graph's node (with higher probability for nodes at lower positions) and following a path in the model until a terminal node is reached (a node without any outgoing edge). A number $N_{noise} = Z * \sum_\sigma |\sigma|$ of noisy random elements (not in the graph) is added to traces, at random positions. We arbitrarily set $Z = 0.1$ (10% of noise in event logs). Resulting traces make a log.

6.1.2. Log Generation for TimeGridOpt (with time)

Logs are generated in the same way as for *GridOpt*, except that edges of generated models are now split in two categories:

- Without time pattern (each transition resulting from this type of edge will have a time between events respecting uniform law $\mathcal{U}(a, b)$);
- With time patterns (a set $\{\mathcal{N}(\mu_i, s_i^2)\}_{i\in\mathbb{N}^*}$ of Gaussian distributions is defined, each transition following one of these laws, randomly selected).

The goal of the optimization is now to highlight temporal patterns in the discovered model with multiple timed edges. The following distributions are used for the log generation:

1. No time pattern $\mathcal{U}(0, 400)$;
2. Simple time pattern ($\{\mathcal{N}(200, s^2)\}$;
3. Double time pattern ($\{\mathcal{N}(100, s^2), \mathcal{N}(300, s^2)\}$;
4. Triple time pattern ($\{\mathcal{N}(100, s^2), \mathcal{N}(200, s^2),$ $\mathcal{N}(300, s^2)\}$);

with s the standard deviation. The allocation of edges to patterns (1-4) is uniform. For double or triple time patterns, the choice among distributions is equiprobable. The value of s is set to 25, to test the robustness of the method for time with variability while keeping patterns identifiable. The edge constraint U_E is set to 80 ($4 \times U_N$) to allow the model to add multiple edges. Other parameters are set as for previous design of experiments. For KDE clustering, $\epsilon = 0.05 \times max(time\text{-}stamp)$ to have a precise time interval for edges.

6.2. Design of Experiments

For each of 15 combinations of ($div_{e,p}$, div_e, p_{max}) used to create logs, 10 random *G-PsM* are created. From each *G-PsM*, a log of 1,000 traces is generated. 5 methods are then applied to solve the problem: Random search (RS), simple Frequency model (FM) obtained by only taking most frequent nodes and edges, Spring search (SS), Tabu search (TS) and Tabu Search with Optimized Edges (TSOE). Figure 5 summarizes the design of experiments at hand. For the search algorithms, stopping criteria are the maximum number of iterations ($x = 250$) or the number of iterations without improvement ($x = 25$). The neighborhood's size for TS and TSOE is empirically set to 15, based on previous tests showing small variability in the best obtained solution's replayability. Similarly, the Tabu list's size is set to 15. The size constraints (number of nodes, number of edges and maximal position) are constant throughout the entire experimental design to always get an interpretative and comprehensive model. Constant parameters and constraint values are summarized in Table 4. Configurations of the design of experiments are listed in the left part of Table 5. For *TimeGridOpt*, configurations 2, 4 and 15 are tested.

6.3. GridOpt Results

The Figure 6 shows the evolution of replayability during the optimization process for each method, specifically applied to configurations 2 and 15. Graphs used to generate event logs are also tested on noised data to compare results of different methods with the initial model used for trace generation, without size limitations ("ROOT" in the figure). Median replayability among 10 event logs for each configuration is presented versus the number of iterations (maximum value for the number of iterations of each method is set to the minimum stopping criterion among 10 replications). ROOT and FM models are obtained by non-iterative methods, their means are displayed by horizontal lines on the same figure for comparison purposes. For RS, an improving solution is rarely obtained, and the stopping criterion is more quickly reached compared to other iterative methods. The margin for improvement during search is small for complex data sets as visible by comparison of Figure 6b and Figure 6a. Furthermore, the gap between ROOT model and search solution strongly increases from Figure 6a to Figure 6b.

Event log description for *GridOpt*, design of experiments and resulting replayability of the best mined models are given in Table 5. Computation times are presented in Table 6. Neighborhood searches (TS and TSOE) systematically outperform other methods (including the heuristic SS and the frequency model FM).

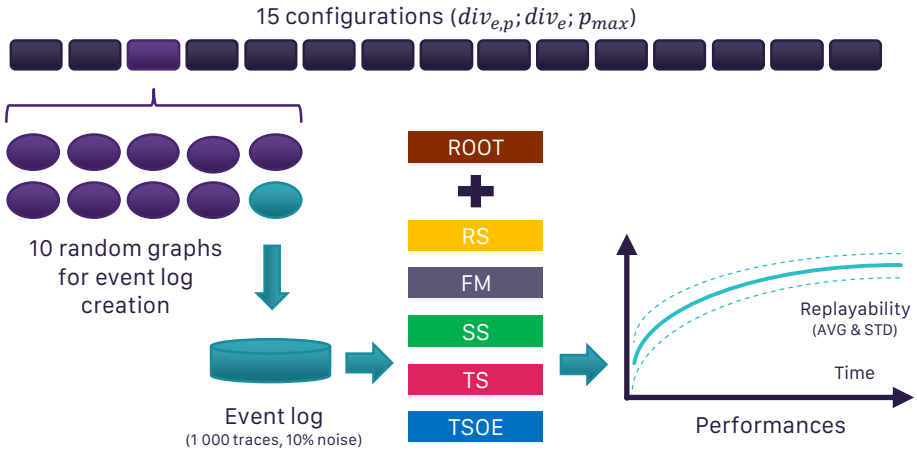


Figure 5: Schematic representation of the design of experiments.

Replayability parameters	
β	0.1
γ	0.1
Search parameters	
K (SS)	2
Neighborhood's size (TS and TSOE)	15
Size of Tabu list	15
Max. number of iterations	250
Max. number of iterations without improvement	25
Constraints	
Number of nodes U_N	20
Number of edges U_E	40 ($2\times U_N$)
Maximal position p_{max}	min(10, $\lvert\sigma\rvert_{max}$)

Table 4: Search parameters and constraints used for design of experiments.

TSOE outperforms TS on 9 out of 15 data configurations, especially when $div_e = 5$. Otherwise, TSOE and TS perform equally. TSOE scores ranges from 0.26 to 0.90. Lower values (< 0.30) are obtained for complex data configurations ($div_{e,p} = 300$ and $div_e = 100$), due to the model size constraints. The unconstrained model used for event log generation (ROOT, where $|N| = div_{e,p}$) systematically scores at 0.90 ± 0.01.

Visual representations of the best models mined by TSOE are presented in Figure 7. Visualization of a process model is possible via a tablet application developed by the company HEVA for that purpose. Each graph is read from left to right, increasing positions. Circles represent nodes of the model, and flux from circles represents edges. The size of nodes and edges are proportional to the number of traces replayed by them during the replayability game. The first qualitative observation is the repetition of events with the same label (Figure 7a), with Label 1 or Label 4. The strong decrease in replayability score from Figure 7a to Figure 7b is visible in the decrease in node and edge size, as fewer traces are well represented. If we focus on edges, Figure 8 highlights this strong decrease. Within the optimization for edges, the leeway in replayability is reduced because of the decrease in the number of patients going through edge pathways (172 vs 38 patients in the example of Figure 8). For this reason, the effect of edge optimization in TSOE is less visible in more complex data sets as Figure 7b compared to Figure 7a.

6.4. TimeGridOpt Results

Results are presented in Table 7. For each data configuration, the mean number of incoherent edges (edges which do not correspond to any defined pattern through design of experiments, by not containing

	Data			RS		FM		SS		TS		TSOE		ROOT	
	$div_{e,p}$	div_e	p_{max}	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
1	25	5	5	0.33	0.08	0.87	0.00	0.86	0.00	0.87	0.00	**0.90**	0.00	*0.91*	*0.00*
2	25	5	25	0.15	0.09	0.77	0.02	0.74	0.02	0.81	0.02	**0.84**	0.02	*0.90*	*0.01*
3	100	5	25	0.24	0.04	0.70	0.01	0.70	0.02	0.75	0.01	**0.77**	0.01	*0.90*	*0.01*
4	100	5	50	0.14	0.08	0.66	0.01	0.65	0.02	0.72	0.01	**0.74**	0.01	*0.89*	*0.01*
5	100	50	5	0.19	0.04	0.34	0.01	0.35	0.02	**0.44**	0.01	**0.44**	0.02	*0.90*	*0.01*
6	100	50	25	0.15	0.04	0.34	0.03	0.33	0.03	**0.42**	0.02	**0.42**	0.02	*0.90*	*0.01*
7	100	50	50	0.11	0.02	0.34	0.04	0.33	0.03	0.41	0.02	**0.42**	0.02	*0.89*	*0.01*
8	100	100	5	0.13	0.02	0.23	0.04	0.25	0.03	**0.36**	0.03	**0.36**	0.02	*0.90*	*0.01*
9	100	100	25	0.13	0.04	0.29	0.03	0.27	0.04	0.37	0.02	**0.38**	0.02	*0.90*	*0.01*
10	100	100	50	0.10	0.03	0.30	0.02	0.29	0.03	**0.38**	0.03	**0.38**	0.02	*0.89*	*0.01*
11	300	50	25	0.15	0.02	0.26	0.02	0.26	0.01	**0.31**	0.01	**0.31**	0.02	*0.89*	*0.01*
12	300	50	50	0.13	0.02	0.24	0.01	0.24	0.02	0.30	0.01	**0.31**	0.01	*0.89*	*0.01*
13	300	100	5	0.13	0.02	0.20	0.01	0.21	0.01	**0.26**	0.01	**0.26**	0.01	*0.89*	*0.01*
14	300	100	25	0.11	0.01	0.21	0.02	0.21	0.02	0.27	0.02	**0.28**	0.02	*0.89*	*0.01*
15	300	100	50	0.10	0.02	0.21	0.02	0.21	0.02	0.27	0.02	**0.29**	0.02	*0.89*	*0.01*

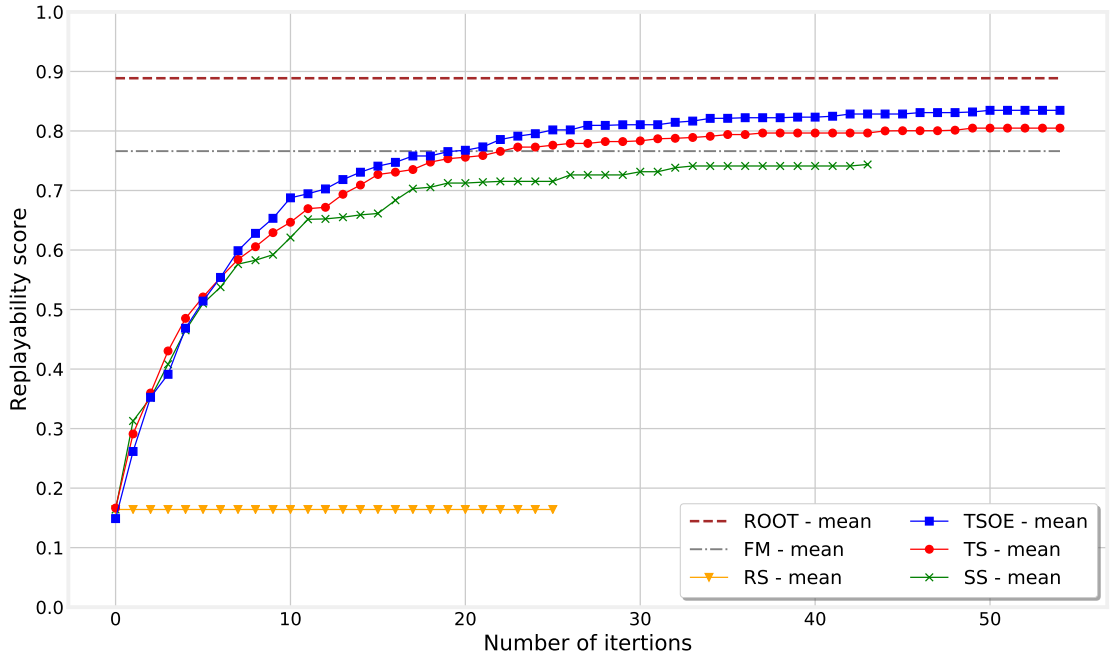
Table 5: The replayability score of the best models mined by different methods: average and standard deviation.

Data	RS		SS		TS		TSOE	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD
1	21	9	39	9	622	151	1479	392
2	15	1	33	6	731	155	1825	419
3	22	7	40	11	665	134	1591	619
4	19	5	38	6	873	234	1930	598
5	34	12	47	16	878	363	1828	362
6	18	5	39	12	711	191	1463	321
7	18	5	33	7	726	162	1541	407
8	15	1	40	12	679	184	1182	335
9	15	<1	33	10	678	162	1513	398
10	15	<1	35	8	689	150	1416	454
11	19	5	38	8	639	99	1353	411
12	18	2	38	8	745	182	1404	334
13	15	<1	37	6	785	213	1354	347
14	16	1	43	8	656	132	1406	253
15	16	<1	36	7	674	173	1497	257

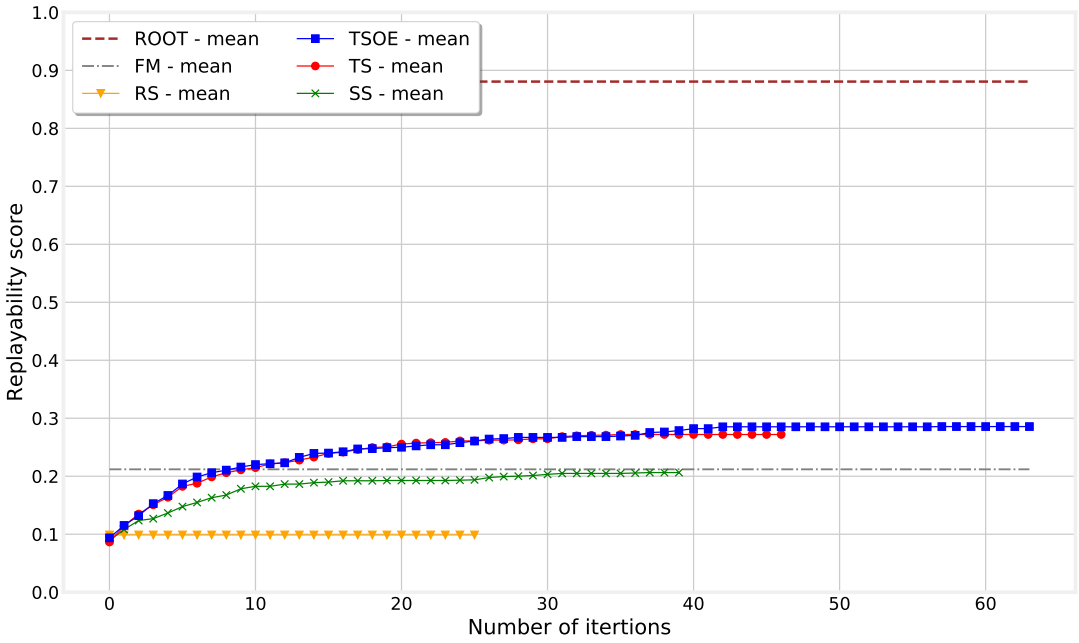
Table 6: Computation time (in seconds) of each method: average and standard deviation.

Data	Replayability		Time		Incoherent edges
Config.	AVG	STD	AVG	STD	AVG
2	0.81	0.02	9820	3048	5.6%
4	0.72	0.01	7965	2547	7.0%
15	0.28	0.02	10999	3688	6.0%

Table 7: Best models mined by TSOE for *TimeGridOpt*: replayability, time (in seconds) and percentage of incoherent edges.

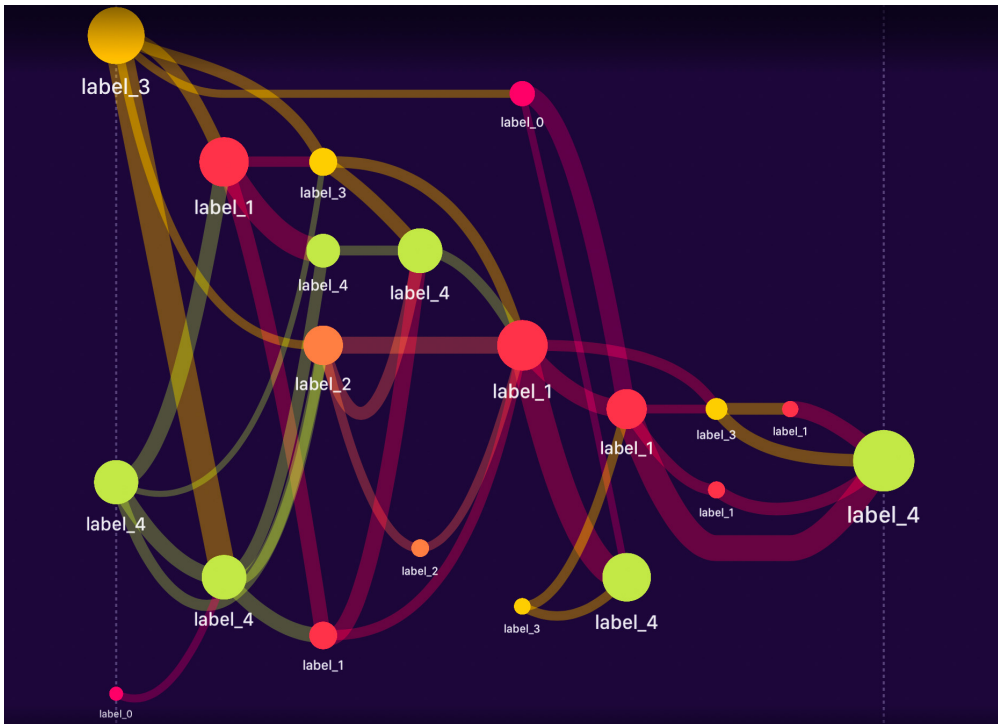


(a) Configuration 2 - GridOpt

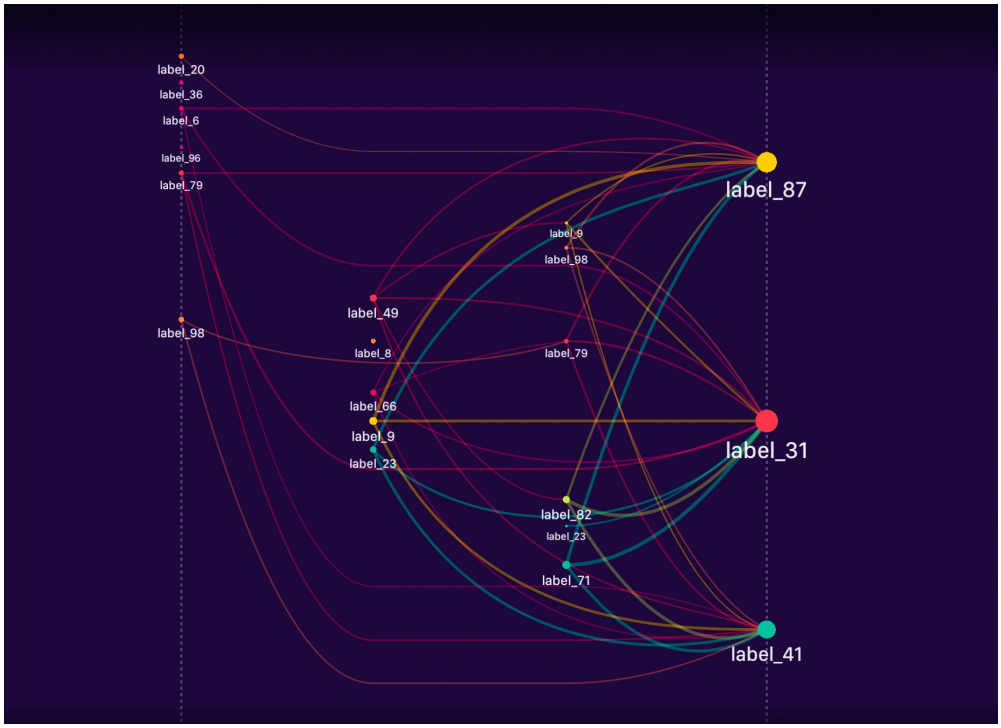


(b) Configuration 15 - GridOpt

Figure 6: Replayability versus the number of iterations: 6 different methods applied to three logs for the *GridOpt* problem; log of configuration 2 (Fig. 6a) and configuration 15 (Fig. 6b).

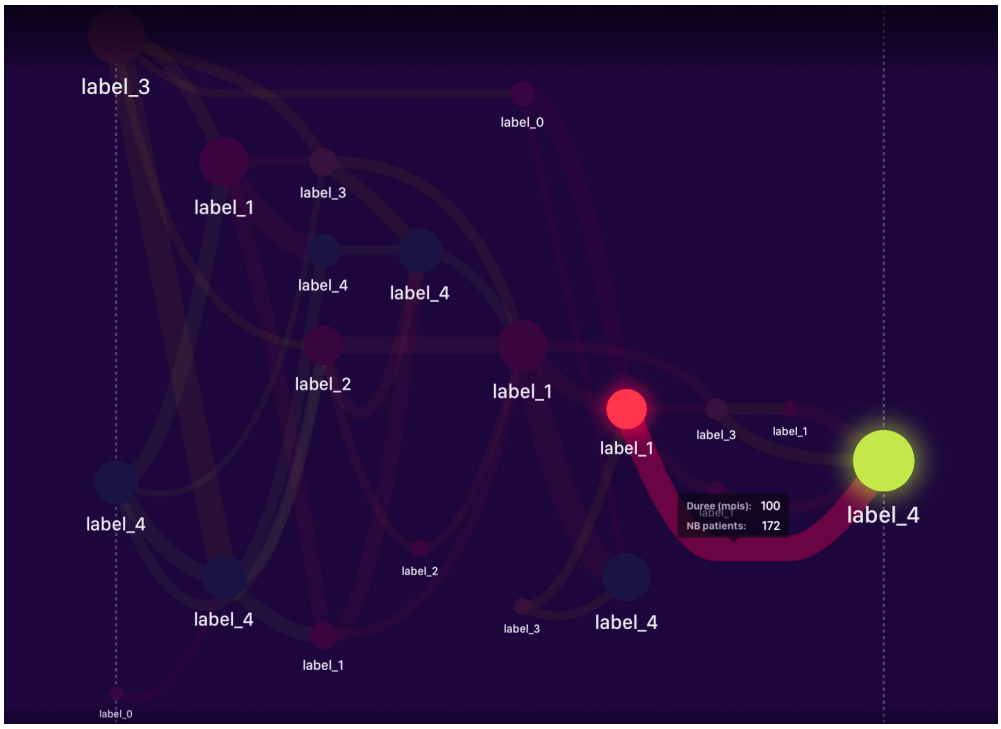


(a) Configuration 2



(b) Configuration 15

Figure 7: Examples of the best *G-PsM* models mined by the TSOE algorithm.



(a) Configuration 2

(b) Configuration 15

Figure 8: Focus on the edges for $G\text{-}PsM$ models.

time values 100, 200 or 300) is also given. The replayability of the best mined models with TSOE for *TimeGridOpt* is slightly inferior to TSOE for *GridOpt* on the same event log. The number of incoherent edges, i.e. the edges not respecting time patterns, is low (5.6%, 7.0% and 6.0%) and thus is encouraging for the methodology presented. These incoherent edges characterize traces with transitions generated from no time pattern edges (25% of transitions following $\{\mathcal{U}(0, 400)\}$) or noisy transitions obtained after adding noise to the event log. Visual representations of the best models are shown in Figure 9 for configuration 2. The general shape of the process model obtained by solving *TimeGridOpt* is similar to previous *G-PsM* graphs (Figure 9a). The time-focused representation of *TG-PsM* highlights the type of edges obtained after the optimization, corresponding to the amount of timed edges it contains. According to the simulated event log, the time pattern edges could be of 3 types: simple (one interval centered in 200), double (2 intervals centered in 100 and 300) or triple (centered in 100, 200 and 300), as shown in Figure 9b.

7. Real-life Case Study

7.1. Diabetes Mellitus

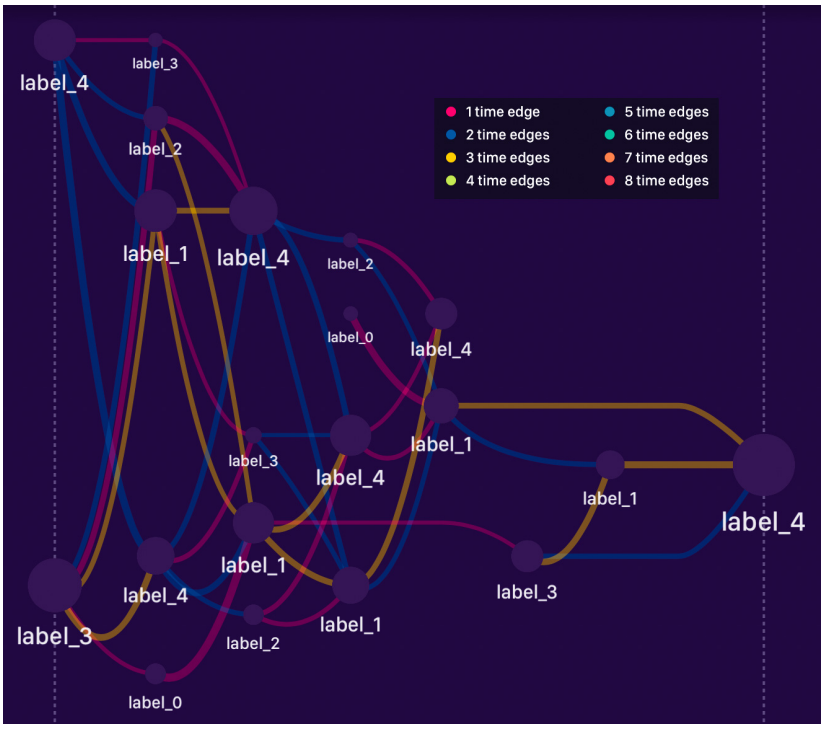
Diabetes Mellitus (DM) is a group of metabolic disorders, resulting in chronic hyperglycemia due to unregulated insulin secretion and/or action. Common forms of DM are type-1 and type-2. Type-2 diabetes is the most common form (90-95% of patients). It is mainly characterized by insulin resistance and relates to the lifestyle, physical activity, dietary habits and heredity. Type-1 diabetes is less frequent (5-10%) and is due to destruction of β cells of the pancreas [4]. Data Mining methods have been widely applied to DM data and supervised learning prevails (85% of studies). Moreover, clinical data sets were the most used [3]. Our method, unsupervised Process Mining, adds a new angle and diversity to existing approaches in DM research. This real-life case study shows how the newly developed approach helps to analyze patient pathways before the appearance of four identified complications.

7.2. Data and Methodology

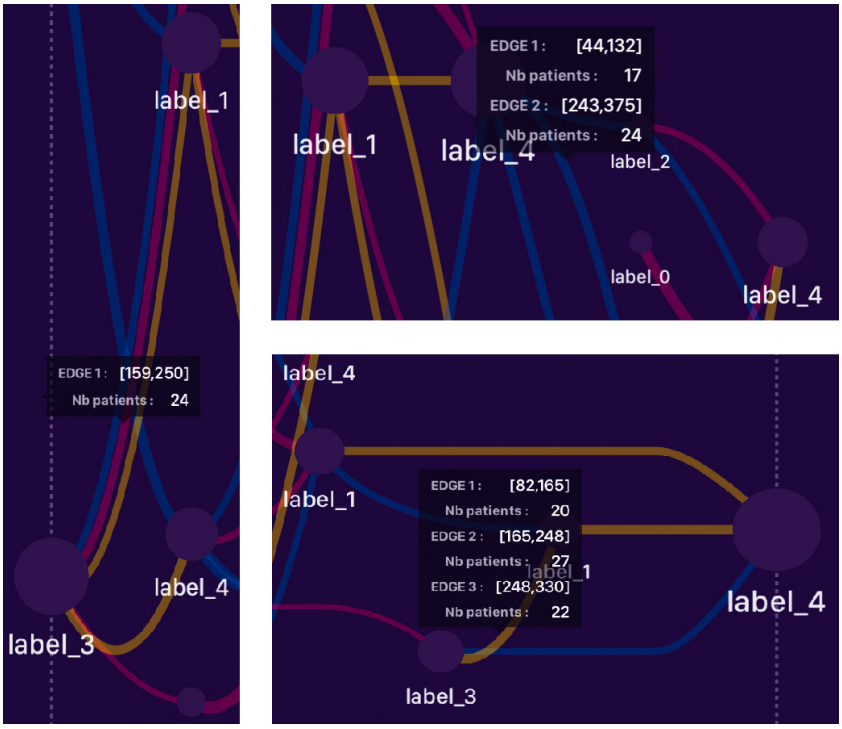
Data originates from the French National Health Insurance (CNAM), where a group of 50,000 patients suffering type-1 or type-2 diabetes in 2008 was constructed. Within this population, 5,714 patients developed at least one of the following complications until 2016: stroke, amputation, infarctus or TCKD (Terminal Chronic Kidney Disease). For each complication and for each patient, a 2-year period of medical history was analyzed. A time grid process model was built for each complication over these 2-year periods. TSOE algorithm was used with the following parameters : models' size is $|N| = 20$, $|E| = 4 \times |N|$, and otherwise as in Table 4. Events of different categories were available:

- Hospitalizations (diabetes, cardiovascular, surgery...);
- Complications (stroke, amputation, infarctus, TCKD);
- Other medical events (dialysis , insulin, emergency without hospitalization).

Other follow-up exams, much more frequent in patient pathways (around 70% of the events), were also available: general practitioner visits, glycated hemoglobin tests (HBA1C), glycemia tests, creatinine tests, etc. Discussions with medical experts led to the non-consideration of these exams as key nodes for the process model. Instead of studying their sequence and successions in the pathway, they were simply and usefully quantified within the period between two nodes (i.e. on an edge). The quantification of such events was performed on the final mined model: for each patient crossing an edge during the replayability game, a list of frequent exams is computed, and median values for each frequent exam are printed on the edges. An unconnected grid process model with best time grid process model's nodes was created first. Then, a number of edges $|E|$, equal to the Div_E, are used to connect the grid process model using optimized edges.



(a) Configuration 2



(b) Focus on the edges for *TG-PsM* models.

Figure 9: Examples of the best *TG-PsM* models mined by the TSOE algorithm

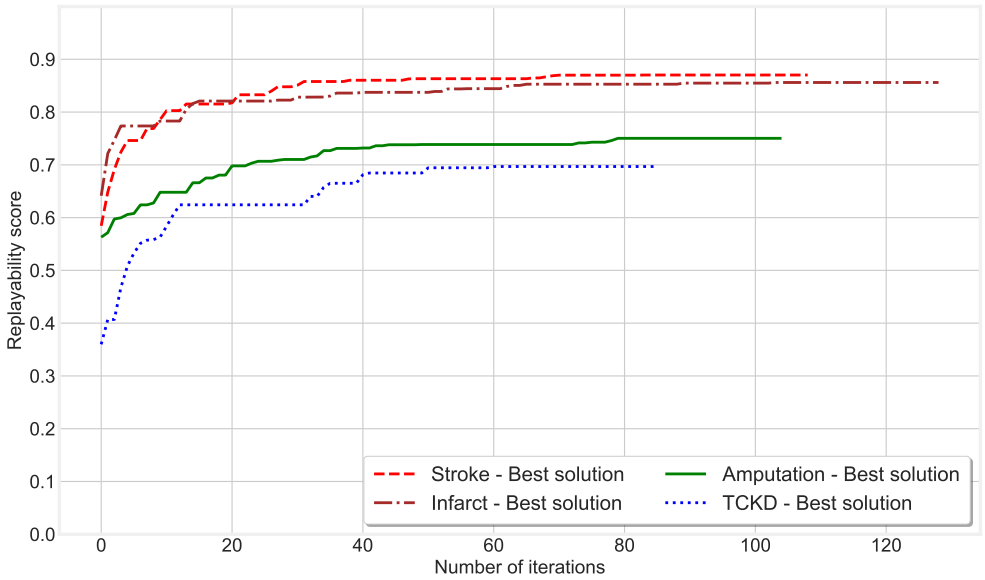


Figure 10: Replayability versus iterations for the four complication event logs.

Complication	$\|L\|$	div_e	$div_{e,p}$	$\|\sigma\|_{q_1,m,q_3}$	R^G	R^{TG}
Amputation	695	15	232	3/5/8	0.76	0.75
Stroke	2152	15	222	1/3/4	0.89	0.87
Infarctus	2913	15	253	2/3/5	0.88	0.86
TCKD	421	13	225	4/6/9	0.72	0.70

Table 8: Diabetes event log descriptors and replayability score.

7.3. Results

Descriptors and replayability performances are presented in Table 8. The evolution of the optimization for each event log is presented in Figure 10. Event log analyses using the descriptors show pathway differences between stroke, infarctus, amputation and TKCD. Indeed, the first two complications are characterized by shorter traces ($|\sigma|_{q_1,m,q_3} : (1,3,4)$ and $(2,3,5)$ vs. $(3,5,8)$ and $(4,6,9)$), that is to say short and unstructured pathways compared to the other two. This difference between complications is also highlighted by the best final replayability scores: amputation and TCKD have lower scores (R^{TG} : 0.75 and 0.70) compared with those of stroke and infarctus (0.87 and 0.86) because longer and more complex pathways are less easily replayed in a graph than shorter ones. These observations are illustrated by Figure 11. An example of frequent events' information can be seen in Figure 11b where a pattern of diabetes hospitalization before stroke is highlighted. For 206 patients concerned, time between events was 242 days on average. As an example of frequent exams, the median number of general practitioner visits is displayed ("MG : 5"). The grid structure, which allows duplicate labels in a process model, is particularly suitable in this case study. As shown in Figure 11a, a high number of "Other hospitalizations excluding surgery" and "Cardiology hospitalizations" are interesting patterns revealed by the grid process model. Time pathway analysis gives further opportunities for understanding patient pathways. As an example, the process model relating to the complication "amputation" (Figure 11a) shows globally unique short time pathways. On the opposite, the process model relating to the complication "stroke" (Figure 11b) presents diverse time pathways, with not only short duration transition.

8. Conclusion and Future Research

An extended methodology to create suitable process models for healthcare applications has been presented. Its Scientific contributions are multiple. New process models considering a grid structure and time patterns were mathematically defined. We formulated a set of descriptors to characterize the structure of such a process model and event log complexity. The establishment of a new property for grid process models leads to a novel search algorithm to mine optimized process models. The search incorporates the grid structure and includes time patterns upon construction of the process model. Computational experiments validate the overall performance of this approach. The interest of neighborhood-based searches to solve the problem was quantitatively shown: Tabu Search with Optimized Edges is more efficient for a small event diversity. A qualitative observation was made regarding the grid structure, representing with more fidelity the linearity of patient pathways over time. This improves the visualization of repeated events. The advantage of considering time within optimization was also spotlighted. In addition, the applicability of the method and the interest in patient pathways analysis is demonstrated by a case study. The grid structure, the time patterns and the display of certain frequent events on edges provide interpretative highlights for medical staff and decision makers.

Three opportunities for future work come to mind. Firstly, a focus on optimization performances for complex data sets will be made, by considering less strict constraints (nodes and edges). During the experiments presented in this work, constraints were specifically set to obtain an overall comprehensible model capable of being simply visually interpreted. However, increasing the complexity of the process model can be achieved if interactive tools permit the exploration of the final model wherein key elements are able to be clearly discerned. Secondly, studying the relation between event log descriptors, graph constraints and replayability of the best models minded is of important interest as well. Any results rendered will be useful for the calibration of constraints, particularly for the third research axis. Eventually, future research should focus on creating a methodology to perform supervised learning with traces as input data, whereas current state-of-the-art classification methods only take "flattened data" as input (vectors of features). A process model optimized for classification purposes will produce an explainable predictive model.

Acknowledgement

The authors wish to thank Chris Yukna for his help in proofreading.

References

[1] T. G. Erdogan and T. Ayca. Systematic mapping of process mining studies in healthcare. *IEEE Access*, 6:1–1, 2018.

[2] A. Giua and X. Xie. Control of Safe Ordinary Petri Nets Using Unfolding. *Discrete Event Dynamic Systems*, 15(4): 349–373, 2005.

[3] I. Kavakiotis, O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda. Machine learning and data mining methods in diabetes research. *Computational and Structural Biotechnology Journal*, 15:104 – 116, 2017.

[4] A. T. Kharroubi and H. M. Darwish. Diabetes mellitus: The epidemic of the century. *World journal of diabetes*, 6: 850–867, 2015.

[5] G. P. Kusuma, M. Hall, C. Gale, and O. Johnson. Process mining in cardiology: A literature review. *International Journal of Bioscience, Biochemistry and Bioinformatics*, 8(4):226–236, 2018.

[6] I. Litchfield, C. Hoye, D. Shukla, R. Backman, A. Turner, M. Lee, and P. Weber. Can process mining automatically describe care pathways of patients with long-term conditions in uk primary care? a study protocol. *BMJ Open*, 8(12), 2018.

[7] A. R. C. Maita, L. C. Martins, C. R. L. Paz, L. Rafferty, P. C. K. Hung, S. M. Peres, and M. Fantinato. A systematic mapping study of process mining. *Enterprise Information Systems*, 12(5):505–549, 2018.

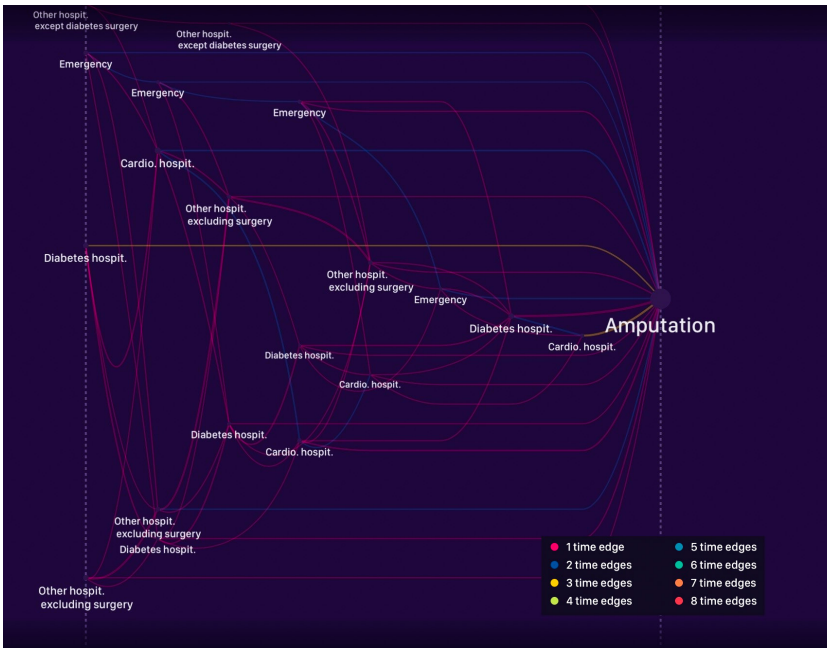
[8] M. Prodel, V. Augusto, X. Xie, B. Jouaneto, and L. Lamarsalle. Discovery of patient pathways from a national hospital database using process mining and integer linear programming. In *CASE*, pages 1409–1414, 2015.

[9] M. Prodel, V. Augusto, B. Jouaneton, L. Lamarsalle, and X. Xie. Evaluation of discovered clinical pathways using process mining and joint agent-based discrete-event simulation. In *Proceedings of the Winter Simulation Conference 2016*, 2016.

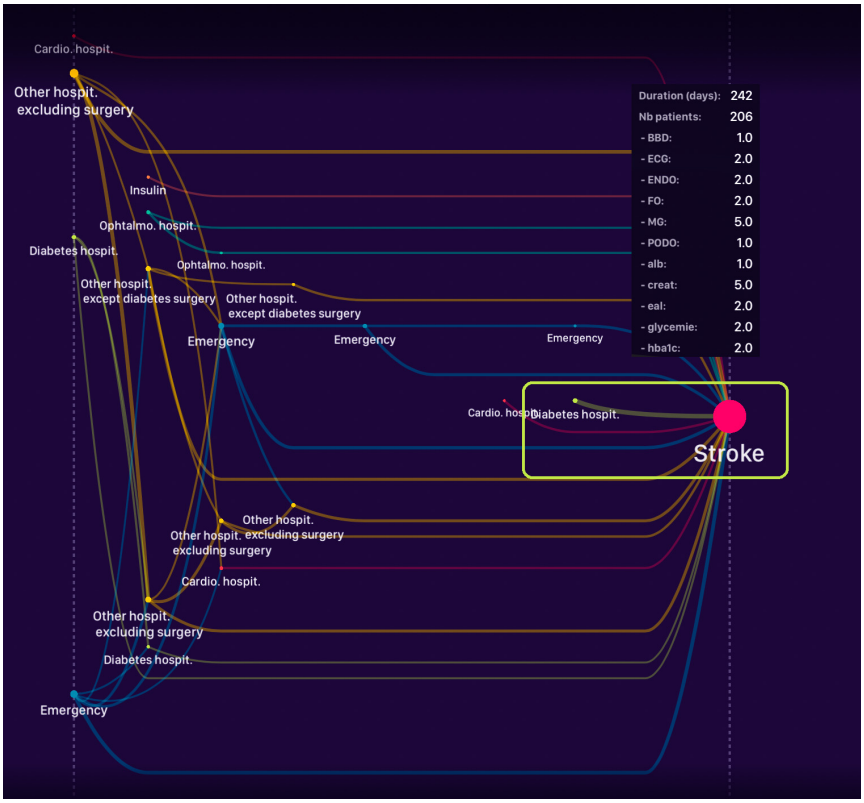
[10] M. Prodel, V. Augusto, B. Jouaneton, L. Lamarsalle, and X. Xie. Optimal process mining for large and complex event logs. *IEEE Transactions on Automation Science and Engineering*, 15(3):1309–1325, 2018.

[11] H. A. Reijers and J. Mendling. A study into the factors that influence the understandability of business process models. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(3):449–462, 2011.

[12] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Springer US, 1986.
[13] W. M. P. van der Aalst. Introduction. In *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, pages 1–25. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
[14] W. M. P. van der Aalst, T. Weijters, and L. Maruster. Workflow mining: discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.



(a) Amputation



(b) Stroke (with an example of frequent events display)

Figure 11: Example of time grid process models resulting from the case study.