Chronicle Discovery for Diagnosis from Raw Data: A Clustering Approach
Alexandre Sahuguède, Euriell Le Corronc, Marie-Véronique V Le Lann

To cite this version:
Alexandre Sahuguède, Euriell Le Corronc, Marie-Véronique V Le Lann. Chronicle Discovery for Diagnosis from Raw Data: A Clustering Approach. 10th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes, SAFEPROCESS 2018, Aug 2018, Warsaw, Poland. 8p. hal-01817529

HAL Id: hal-01817529
https://hal.laas.fr/hal-01817529
Submitted on 18 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Chronicle Discovery for Diagnosis from Raw Data: A Clustering Approach

Alexandre Sahuguède∗ Euriell Le Corronc∗ Marie-Véronique Le Lann∗

∗ LAAS-CNRS, Université de Toulouse, CNRS, INSA, UPS, Toulouse, France (e-mail: {asahugue, elecorro, mvlelann}@laas.fr)

Abstract: Chronicles are temporal patterns well suited for an abstract representation of the behavior of dynamic systems. For fault diagnosis, chronicles describe the nominal and faulty behaviors of the process. Powerful algorithms allow the recognition of chronicles in the flow of observations of the system and appropriate actions can be taken when a faulty situation is recognized. However, designing chronicles is not a trivial thing to do. The increasing complexity and capacity of data generation of highly-advanced processes cause the acquisition of a complete model difficult. This paper focuses on the problem of discovering chronicles that are representative of a system behavior from direct observations. A clustering approach to this problem is considered. The chronicle discovery algorithm proposed here designs chronicles with minimal knowledge of the system to diagnose. Furthermore, unprocessed data obtained directly from the system can be used in this clustering algorithm. Finally, the chronicle discovery algorithm proposed in this paper is illustrated on a sport performance monitoring device for a diagnosis of movement deviations in the temporal domain, in the event domain, or both, considered as faults for the athlete.

Keywords: Fault Diagnosis, Machine Learning, Clustering, Temporal Pattern Mining, Chronicle Discovery.

1. INTRODUCTION

This paper focuses on the fault diagnosis of dynamic complex systems problem. The problem consists in producing a temporal model well suited for fault diagnosis from a set of observations of a complex process. The increasing complexity of modern systems and the development of data generation and storage capacities greatly increase the number of observations. This large number of observations should be processed by some automatic method in order to assist the expert in the design of such a temporal model.

The studied models are timed discrete event models called chronicles (Dousson and Le Maigat, 2007). They are temporal patterns well suited to capture the behavior of a dynamic process. Chronicles describe behaviors by means of an event abstraction of the information of interest. Within this formalism, events are partially ordered and temporally constrained one to another. Figure 1 describes such a chronicle where events of interest are $a$, $b$ and $c$. The illustrated process is such that an event of type $c$ must occur between 3 and 4 time units ($t.u.$) after an event of type $b$. This event must appear between 1 and 2 $t.u.$ later than an event of type $a$. Another event of type $c$ must follow $a$ anytime bounded by 8 and 10 $t.u.$. In the diagnosis domain, each chronicle represents a specific behavior of the system. This behavior can be nominal or typical when a fault is present. This faulty behavior can be recognized in a temporal sequence made of timed observations generated by the system. The recognition of this specific temporal pattern leads to establish that the system is in that faulty situation (Dousson and Le Maigat, 2007). Various diagnosis applications use the chronicle approach. A chronicle based diagnosis in web services is presented in (Pencolé and Subias, 2009). A multi-alarm misuse correlation component that allows the user to significantly reduce the number of alarms uses a chronicle approach in (Morin and Debar, 2003). Chronicles could be used in the medical field as for instance in (Carrault et al., 1999) where they allow identification of cardiac arrhythmia and in (Dauxais et al., 2017) where possible associations between hospitalizations for seizure and anti-epileptic drug switches are identified. However, as efficient a diagnosis with chronicles can be, designing chronicles is not a trivial thing to do. The design of such a complex model often requires the knowledge of an expert of the process to be diagnosed. Unfortunately, this knowledge is frequently not enough to build pertinent chronicles for diagnosis.

![Fig. 1. A chronicle where events of interest are $a$, $b$ and $c$.](image)

The Apriori algorithm (Agrawal and Srikant, 1994) is a commonly used approach to try and tackle this problem. Apriori is a data mining algorithm that find frequent collections of items in sale transactions. It finds iteratively
larger and larger itemsets. An itemset found to be frequent in size \( k \) will be discarded if a sub-itemset of size \( k-1 \) is not frequent. This Apriori algorithm can be used for sequential pattern mining (Mannila et al., 1995) as well as temporal pattern mining (Guyet and Quiniou, 2011). Several chronicle discovery algorithms are based on the Apriori algorithm as in (Cram et al., 2012). In his paper, Cram presents a chronicle discovery algorithm from a temporal sequence. This algorithm is extended to the multi-sequences case by (Subias et al., 2014). First, it builds a database of time constraints. Then, it generates a set of candidate chronicles starting with a set of chronicles that were proved to be frequent. A temporal constraint network discovery algorithm is presented in (Álvarez et al., 2013) and uses a clustering algorithm reducing the set of candidates. This algorithm could be convenient for chronic discovery as temporal constraint networks are similar to chronicles.

The main limitation of these Apriori-based approaches lies in the fact that a minimum frequency parameter is necessary. The choice of this parameter is not trivial and requires a good amount of knowledge of the system to obtain good results. Another limitation is that these algorithms are done from one, or several, temporal sequences, meaning that only observations of the system made of events can be used. Unfortunately, very often observations of the dynamic process behaviors are composed of sampled continuous signals obtained from sensor measurements, logs from communication networks or healthcare data.

This paper tries to offer another solution with a clustering approach applied directly to raw data. Machine learning techniques and more specifically clustering, based on the density of the data such as DBSCAN (Density Based Spatial Clustering and Application with Noise (Ester et al., 1996)) or on the fuzzy logic such as LAMDA (Learning Algorithm for Multivariate Data Analysis (Carreté and Aguilar-Martin, 1991)), are applied to discover chronicles. First, a temporal sequence is extracted from the raw data provided by the system observations. Then, clustering techniques are used to regroup some patterns by similarity. A strong advantage of this method is that the frequency of the pattern found is deduced from the data. Clusters found are considered as the most representative patterns of the system. Chronicles discovered by the proposed algorithm are various in length (the number of events of a chronicle) and in frequency. They can be recognized on-line by a chronicle recognition algorithm. Chronicles obtained are an abstract representation of the dynamic system behavior which describes either the nominal or a faulty behavior. Contrary to the exponential algorithmic complexity of the Apriori approach, the proposed method can be done with a polynomial algorithmic complexity.

The rest of this paper is organized as follows. In Section 2, definitions of required concepts are explained. Section 3 presents the algorithm that discovers chronicles from raw data. Section 4 introduces an example with a swimming performance monitoring equipment for a health monitoring of the athlete movements. Section 5 concludes this work.

### 2. CONCEPTS AND DEFINITIONS

#### 2.1 Chronicles

This section explains the chronicle concepts used in this work. Chronicles are ways of expressing relevant temporal patterns about a process (Dousseon and Le Maigat, 2007).

**Definition 1.** (Event). An event is defined by \( x = (e, t) \) with an event type \( e \in E \), and a time instant \( t \in \mathbb{N} \).

**Definition 2.** (Temporal sequence). A temporal sequence is a time-ordered set of events denoted \( S = \{x_i\} \) where \( i \in \mathbb{N}, i = 1, \ldots, n \) with \( n \) a finite number of events, and \( t_j < t_{j+1}, j = 1, \ldots, n - 1 \). The set of all event types occurring in \( S \) is called \( E_S \).

**Example 1.** The temporal sequence \( S = \{x_1, x_2, x_3, x_4, x_5\} \) where \( x_1 = (a, 1), x_2 = (b, 2), x_3 = (a, 12), x_4 = (a, 18), \) and \( x_5 = (b, 20) \), graphically represented in Figure 2, points out the difference between event and event type. That is several events can share the same event type. For instance, events \( x_1, x_3 \) and \( x_4 \) are different occurrences of event type \( a \).

#### 2.2 Algorithmic definitions

**Definition 3.** (Chronicle constraint). A temporal constraint is a tuple \( \tau_{ij} = \{x_i, x_j, t^-, t^+\} \), also noted \( \tau_{ij} = x_i[t^-, t^+]x_j \) where \( 0 < t^- \leq t^+ \). A temporal constraint \( \tau_{ij} = x_i[t^-, t^+]x_j \) is said satisfied by a couple of events \( x_i = (e_i, t_i), x_j = (e_j, t_j) \) if and only if \( (t_j - t_i) \in [t^-, t^+] \).

**Definition 4.** (Chronicle). A chronicle \( C \) is a pair \((X, T)\) where \( X = \{x_i\} \) is a set of partially ordered events with \( i \in \mathbb{N}, i = 1, \ldots, n \), and \( n \) a finite number of events, and \( T = \{\tau_{ij}\}_{1 \leq i < j \leq n} \) is a set of temporal constraints on \( X \). \( E_C \) denotes the set of all event types of \( C \). A \( n \)-length chronicle is a chronicle with \( n \) events.

**Example 2.** Let \( C = (X, T) \) be a 4-length chronicle with \( X = \{x_1 = (a, t_1), x_2 = (b, t_2), x_3 = (c, t_3), x_4 = (c, t_4)\} \), and \( T = \{\tau_{12} = x_1[1, 2], \tau_{23} = x_2[3, 4], \tau_{14} = x_1[8, 10], \tau_{43} = x_3[18, 20]\} \). The set of all event types of \( C \) is \( E_C = \{a, b, c\} \).

Graphically, a chronicle is a directed graph where the nodes represent the events of \( X \) and the transitions represent the temporal constraints of \( T \). In each temporal constraint \( \tau_{ij}, x_i \) is the starting node and \( x_j \) is the ending node. For instance, Figure 1 is the graphical representation of the chronicle described in Example 2.

The frequency criterion is chosen to define the relevance of a pattern required for the learning chronic process. Such criterion needs definitions in this domain.

**Definition 5.** (Chronicle instance). Given a chronicle \( C = (X, T) \) and a temporal sequence \( S \), a chronic instance is a subset of events of \( S \) denoted \( f_C(S) \) such that their event types are those of \( X \) and their time occurrences satisfy all the temporal constraints \( T \) of \( C \).

**Definition 6.** (Frequency of a chronicle). The frequency of a chronicle \( C \) in a temporal sequence \( S \) is the number of instances of \( C \) in \( S \) and is named \( f_C(S) \).
Definition 7. (Coherent chronicle). A chronicle $C$ is called coherent if there exists a temporal sequence $S$ such that $f_C(S) > 0$. A coherent chronicle has at least one coherent instance $I_C(S)$.

Example 3. Let us define a chronicle $C = (X, T)$ with $X = \{x_1 = (a, t_1), x_2 = (b, t_2)\}$, and $T = \{t_1 = x_1[1, 2], t_2 = x_2[1, 2]\}$. The frequency $f_C(S)$ for this coherent instance $I_C(S)$ is $0$. Additionally, the occurrence $O_C$ of $I_C(S)$ is all the occurrences of $O_C$ such that $O_C$ follows $a$: $O_C = \{(a, t_i), (b, t_j)\} \forall i, j, t_i < t_j, (a, t_i), (b, t_j) \in S$.

Definition 9. (Temporal distances of a pair). The set $D_{ab}$ is the set of temporal distances between each occurrence of the pair $(a, b)$:

$$D_{ab} = \{(t_j - t_i) \mid ((a, t_i), (b, t_j)) \in O_{ab}\}.$$

Example 4. Given the temporal sequence $S$ of Figure 2, the set $O_{ab}$ is all the temporal distances between each occurrence of the pair $(a, b)$:

$$O_{ab} = \{(a, t_1), (b, t_2)\}.$$  

Proposition 1. If $D_{ab}$ is a set of distances for a pair $(a, b)$, a 2-length chronicle $C = (X, T)$ which can be obtained from $D_{ab}$, $X = \{x_1 = (a, t_1), x_2 = (b, t_2)\}$ is given by the elements of the pair $(a, b)$ and $T = \{t_1 = x_1[1, 2], D_{ab}[x_1] \}$ is given by the lower and upper bounds of $D_{ab}$.

Proof. Directly from Definitions 4 and 9.

All instances $I_C(S)$ of chronicles extracted from the set of temporal distances $D_{ab}$ correspond to the set of their occurrences $O_{ab}$. The frequency $f_C(S)$ is the size of $D_{ab}$.

Example 5. Let $(a, b)$ be a pair of event types, the set of temporal distances calculated in Example 4 is $D_{ab} = \{1, 19, 8, 2\}$ and defines the following 2-length chronicle: $C = (X, T)$ where $X = \{x_1 = (a, t_1), x_2 = (b, t_2)\}$, and $T = \{t_1 = x_1[1, 2], t_2 = x_2[1, 2]\}$. Furthermore, the instances of $C$ in the original temporal sequence $S$ are $I_1(T) = \{(a, 1), (b, 2)\}$, $I_2(T) = \{(a, 1), (b, 20)\}$, $I_3(T) = \{(a, 12), (b, 20)\}$, and $I_4(T) = \{(a, 18), (b, 20)\}$. These instances correspond to the occurrences of the pair $(a, b)$.

2.2 Clustering

This section presents the clustering algorithms required in this work: LAMDA and DBSCAN. They group data samples based on the similarity or dissimilarity of the measures. The distinctive characteristics of data samples are called features while the specific pattern detected in the data are called classes. In this paper, features of interest are issued from on-line sensor measurements of dynamic process, logs of networks communications or healthcare data.

The fuzzy logic based algorithm called LAMDA (Carretil and Aguilar-Martin, 1991) takes as input a sample $k$ made up of $N$ features. Its first step computes for each feature $k$ an adequacy degree to each class, indexed by $1 \ldots J$. With the help of a fuzzy adequacy function, $j$ vectors of $N$ adequacy degrees are computed. They are called Marginal Adequacy Degree vectors (MAD) and can be calculated by different means (Gaussian, fuzzy binomial, centered fuzzy binomial). In a second step, a fuzzy aggregation function assembles all the MADs for a specific class into one Global Adequacy Degree (GAD). This fuzzy function has a parameter $\alpha$, called exigency index. $\alpha$ is given in the $[0, 1]$ interval. The $\alpha$ parameter has a direct impact on the number of classes found: the higher $\alpha$ is, the higher the number of classes found. The $J$ MAD vectors, composed of $N$ MADs, become $J$ scalar GADs. The higher the GAD, the better the adequacy to the class. The simplest way to assign the sample $k$ to a class is to keep as result the class with the biggest GAD.

The density-based clustering algorithm DBSCAN (Ester et al., 1996) works in two steps. First, every density-reachable points from each point of the dataset to be classified are calculated. Second, a random starting point is selected. If it is a core point, meaning that it has in its neighborhood of radius $\epsilon$ at least the minimal number of points $\text{minPts}$, a new cluster is created. Otherwise, this point is determined as noise. This point and every points density-reachable from it with $\epsilon$ is added in the created cluster. This cluster is then expanded by selecting the new density-reachable point from the core point and determining if it is another core point. Then, its density-reachable points are added in the cluster. When the cluster is fully expanded, a new unvisited point is retrieved and this step is repeated, leading to the discovery of new clusters and noises.

3. CHRONICLE DISCOVERY FROM RAW DATA

As presented in Figure 3, the chronicle discovery algorithm proposed in this paper works in several steps. First, temporal patterns are discovered from time-ordered raw data. These data are generated by the dynamic system we want to diagnose. Data come from sensors measurements, alarm logs, healthcare data... In a complex dynamic system, unprocessed data can be corrupted by noises of various origins. Those noises are dealt with by the fuzzy logic based clusterer LAMDA. A temporal sequence is extracted from classes by defining events as the changes of classes over time. Next, 2-length chronicles are discovered by the
density based clustering algorithm DBSCAN. It groups
them by similar temporal distances between events. This
clustering step allows to find the frequencies in the tempo-
ral sequence of those 2-length chronicles. Finally, a similar
frequency criterion is used on the 2-length chronicles found
in the previous step to unified them. This is done by
means of the computation of a Jaccard index on the events.
The algorithm provides chronicles of various lengths and
frequencies. Each chronicles representing time patterns
where events are abstraction of the most relevant elements
of the dynamic process. These elements are temporally
constrained to each other.

\begin{align*}
\text{Raw data } R & \\
1. \text{Temporal sequence extraction} & \\
\text{Temporal sequence } S & \\
2. \text{2-length chronicle discovery} & \\
\text{2-length chronicles } C_i & \\
3. \text{Chronicle synthesis} & \\
\text{Learned chronicles } C &
\end{align*}

Fig. 3. An overview of the chronicle discovery from raw
data algorithm. Inputs are time-ordered raw data
taken from the dynamic system. Output is a set of
chronicles describing elements of its behavior.

3.1 Step 1: Temporal sequence extraction

The first step of the algorithm is the temporal sequence
extraction process given in Algorithm 1.

First, totally time-ordered raw data denoted \( R \) are clas-
sified by LAMDA. Since it is a clustering operation, the
number of classes is not known in advance and the choice
of the \( \alpha \) parameter is left to the expert. Once each sample
of the raw data has been assigned to a class, a temporal
sequence \( S \) of this classification is obtained by conser-
ving the time-ordered information. Let us define \( \delta \), a threshold
that represents significant changes in the feature values
over time. An event \( x = (\text{class}_i, t_i) \) is created and added to
\( S \) when the class of the sample \( i \) is different than all the
previous samples \( i - \delta \). The set \( E_S \) of event types of \( S \)
corresponds to the name of the classes and the time instants
are the sample times. Classes with too few consecutive
samples are explained as noise and are discarded.

Example 6. Let us deal with a process with two recogniz-
able behaviors. They are accurately described by the
measurements of two sensors. An evolution of those sensors
over a duration of 1100 t.u. with a sampling rate of one
sample by t.u. is compiled in the dataset \( R \). This dataset
has two features that are interpreted as the measurements
of the sensors. \( R \) is illustrated in Figure 4. The LAMDA
clusterer discriminates two behaviors in the dataset by
taking \( \alpha = 0.2 \) and \( \delta = 1 \). Each behavior corresponds to a
class: class \( e \) with feature 1 decreasing and feature 2 equal
to 0; class \( f \) with feature 1 increasing and feature 2 equal to

\begin{algorithm}
1. INPUT: totally time-ordered raw data \( R \)
2. OUTPUT: temporal sequence \( S \)
3. INIT \( S \) at empty
4. COMPUTE classes of \( R \) with LAMDA algorithm
5. FOR each data sample \( i \) of \( R \)
6. IF the class of \( i \) is different than all the classes of \( i - 1 \) to \( i - \delta \) THEN
7. ADD event \( x = (\text{class}_i, t_i) \) in \( S \)
8. ENDF.
9. ENDFOR.

Algorithm 1. (Temporal sequence extraction).
\end{algorithm}

1. The temporal sequence extracted from this classification
is \( S = \{(e, 0), (f, 98), (e, 202), (f, 700), (e, 798)\} \) with its set
of event types given by \( E_S = \{e, f\} \).

Fig. 4. Dynamic dataset \( R \) with two features, feature 1
in black and feature 2 in gray, corresponding of two
sensors in a dynamic process.

3.2 Step 2: 2-length chronicle discovery

A temporal sequence \( S \) is then generated from the time-
ordered raw data \( R \) thanks to the previous step. One
would like to know if it is possible to gather events of \( S \)
according to some criterion by mean of another clustering
phase. Indeed, by calculating the temporal distances \( D \)
(see Definition 8 and Equation (2)) of all the pairs of events
of \( S \), one can see that some distances of the same set are
close whereas others are further apart. Thus, this step of
the algorithm uses the density based algorithm DBSCAN
to group such similar distances in clusters. During this
clustering phase, \( D \) is interpreted as a dataset with one
dimension feature space.

Then, by using Proposition 1, the algorithm builds 2-
length chronicles \( C \) from the clusters found. This is done by
taking the minimum and the maximum distances of \( D \) for
the temporal constraint of \( C \). The event types of \( C \) are
the event types of \( D \). Since DBSCAN results are homogeneous
clusters, meaning that there is no temporal distance that
satisfies the temporal constraint that is not in \( D \), the
frequency \( f_C(S) \) of the 2-length chronicle is exactly the
number of temporal distances in \( D \).

The minimum frequency of the created 2-length chronicles
depends on the minimum number of points \( minPts \) in the
neighborhood. More precisely, this minimal frequency is
equal to \( minPts + 1 \). The radius parameter \( \varepsilon \) defines the
dispersion of the temporal constraint, when \( \varepsilon \) grows, the
dispersion grows.

Proposition 2. A 2-length chronicle \( C \) designed from a set
of temporal distances \( D \) itself obtained from a temporal
sequence \( S \) is coherent.

Proof. Clusters found by DBSCAN are not empty, oth-
ervise, they would be considered as noise. As clusters are

\begin{align*}
\text{Temporal sequence } S & \\
2-length chronicles } C_i & \\
3. Chronicle synthesis & \\
\text{Learned chronicles } C &
\end{align*}

Fig. 3. An overview of the chronicle discovery from raw
data algorithm. Inputs are time-ordered raw data
taken from the dynamic system. Output is a set of
chronicles describing elements of its behavior.

3.1 Step 1: Temporal sequence extraction

The first step of the algorithm is the temporal sequence
extraction process given in Algorithm 1.

First, totally time-ordered raw data denoted \( R \) are clas-
sified by LAMDA. Since it is a clustering operation, the
number of classes is not known in advance and the choice
of the \( \alpha \) parameter is left to the expert. Once each sample
of the raw data has been assigned to a class, a temporal
sequence \( S \) of this classification is obtained by conser-
ving the time-ordered information. Let us define \( \delta \), a threshold
that represents significant changes in the feature values
over time. An event \( x = (\text{class}_i, t_i) \) is created and added to
\( S \) when the class of the sample \( i \) is different than all the
previous samples \( i - \delta \). The set \( E_S \) of event types of \( S \)
corresponds to the name of the classes and the time instants
are the sample times. Classes with too few consecutive
samples are explained as noise and are discarded.

Example 6. Let us deal with a process with two recogniz-
able behaviors. They are accurately described by the
measurements of two sensors. An evolution of those sensors
over a duration of 1100 t.u. with a sampling rate of one
sample by t.u. is compiled in the dataset \( R \). This dataset
has two features that are interpreted as the measurements
of the sensors. \( R \) is illustrated in Figure 4. The LAMDA
clusterer discriminates two behaviors in the dataset by
taking \( \alpha = 0.2 \) and \( \delta = 1 \). Each behavior corresponds to a
class: class \( e \) with feature 1 decreasing and feature 2 equal
to 0; class \( f \) with feature 1 increasing and feature 2 equal to

\begin{algorithm}
1. INPUT: totally time-ordered raw data \( R \)
2. OUTPUT: temporal sequence \( S \)
3. INIT \( S \) at empty
4. COMPUTE classes of \( R \) with LAMDA algorithm
5. FOR each data sample \( i \) of \( R \)
6. IF the class of \( i \) is different than all the classes of \( i - 1 \) to \( i - \delta \) THEN
7. ADD event \( x = (\text{class}_i, t_i) \) in \( S \)
8. ENDF.
9. ENDFOR.

Algorithm 1. (Temporal sequence extraction).
\end{algorithm}

1. The temporal sequence extracted from this classification
is \( S = \{(e, 0), (f, 98), (e, 202), (f, 700), (e, 798)\} \) with its set
of event types given by \( E_S = \{e, f\} \).

Fig. 4. Dynamic dataset \( R \) with two features, feature 1
in black and feature 2 in gray, corresponding of two
sensors in a dynamic process.

3.2 Step 2: 2-length chronicle discovery

A temporal sequence \( S \) is then generated from the time-
ordered raw data \( R \) thanks to the previous step. One
would like to know if it is possible to gather events of \( S \)
according to some criterion by mean of another clustering
phase. Indeed, by calculating the temporal distances \( D \)
(see Definition 8 and Equation (2)) of all the pairs of events
of \( S \), one can see that some distances of the same set are
close whereas others are further apart. Thus, this step of
the algorithm uses the density based algorithm DBSCAN
to group such similar distances in clusters. During this
clustering phase, \( D \) is interpreted as a dataset with one
dimension feature space.

Then, by using Proposition 1, the algorithm builds 2-
length chronicles \( C \) from the clusters found. This is done by
taking the minimum and the maximum distances of \( D \) for
the temporal constraint of \( C \). The event types of \( C \) are
the event types of \( D \). Since DBSCAN results are homogeneous
clusters, meaning that there is no temporal distance that
satisfies the temporal constraint that is not in \( D \), the
frequency \( f_C(S) \) of the 2-length chronicle is exactly the
number of temporal distances in \( D \).

The minimum frequency of the created 2-length chronicles
depends on the minimum number of points \( minPts \) in the
neighborhood. More precisely, this minimal frequency is
equal to \( minPts + 1 \). The radius parameter \( \varepsilon \) defines the
dispersion of the temporal constraint, when \( \varepsilon \) grows, the
dispersion grows.

Proposition 2. A 2-length chronicle \( C \) designed from a set
of temporal distances \( D \) itself obtained from a temporal
sequence \( S \) is coherent.

Proof. Clusters found by DBSCAN are not empty, oth-
ervise, they would be considered as noise. As clusters are

not empty, the set of temporal distances $D$ are also not empty. The frequency of the designed chronicles $f_C(S)$ is more than 1. The created 2-length chronicle is coherent.

**Algorithm 2. (2-length chronicle discovery).**

1. INPUT: temporal sequence $S$
2. OUTPUT: 2-length chronicles $C_1$
3. FOR each pair of event types $a$ and $b$ of $E_S$
4. CALCULATE temporal distances $D_{ab}$ in $S$
5. CALCULATE clusters in $D_{ab}$ with DBSCAN
6. FOR each clusters found $D_{ab}$
7. TRANSFORM $D_{ab}$ in a 2-length chronicle
8. ENDFOREACH
9. ENDFOR

The 2-length chronicles discovery algorithm is presented in Algorithm 2 given a temporal sequence $S$. First, for each pair of event types from $E_S$ (called $a$ and $b$ for the explanation of this algorithm, but pair $a$ and $a$ is also taken), the set of temporal distances $D_{ab}$ is calculated by Equation (2). A cluster analysis is performed on $D_{ab}$ with the DBSCAN algorithm. Finally, each cluster $D_{ab}$ found defines a 2-length chronicle.

**Example 7.** Let the temporal sequence generated in Example 6 be $S = \{(e,0), (f,98), (e,202), (f,700), (e,798)\}$ with $E_S = \{(e,f)\}$. First, let us see the pair $(f,e)$. Algorithm 2 provides $D_{fe} = \{104,700,98\}$ as the set of all temporal distances for this pair. The DBSCAN parameters are set such that $minPts = 1$, and $\varepsilon = 7$. As a result, only one cluster is found: $D_{fe} = \{104,98\}$. The remaining temporal distance is in a low density area and is considered irrelevant. For the other pairs $(e,e)$, $(e,f)$, and $(f,f)$, the clustering algorithm does not find temporal distances close enough to group them. The chronicle $C_1$ is obtained from $D_{fe}$ using Proposition 1 and is illustrated in Figure 5. It is a 2-length chronicle with $X_1 = \{x_1 = (f,t_1), x_2 = (e,t_2)\}$, and $T_1 = \{\tau_{12} = x_1[98,104],x_2[98,104]\}$. The frequency of $C_1$ in $S$ is $f_C(S) = 2$ and its two instances in $S$ are: $T_1 = \{(f,98),(e,202)\}$, and $T_2 = \{(f,700),(e,798)\}$. With Proposition 2, $C_1$ is coherent.

This index first needs instances of created chronicles to find identical occurrences of events obtained by different 2-length chronicles of the same frequency.

**Definition 10.** (Time occurrences of an event). Let $S$ be a temporal sequence and $C = (X,T)$ be a chronicle. $O_i$ is the set of time occurrences of the event $x_i$ in all chronicles instances $I_i(S)$. $O_i$ is calculated by the following formula:

$$O_i = \{t_i \mid \forall t \in I_i(S), x_i = (e,t_i) \in X\}. \quad (3)$$

The size of $O_i$ is given by $f_C(S)$ and is denoted $|O_i|$.

**Example 8.** Given the chronicle $C$ and the temporal sequence $S$ seen in Example 3, the two chronicles instances $T_1 = \{(f,98),(e,202)\}$ and $T_2 = \{(f,700),(e,798)\}$ give the two occurrences of $x_1$: $(a,1)$ and $(a,18)$. The set of time occurrences of $x_1$ is $O_1 = \{1,18\}$ ($(|O_1| = 2$). For $x_2$, $O_2 = \{2,20\}$ and $|O_2| = 2$.

**Definition 11.** (Jaccard index). Let $x_i$ and $x_j$ be two events with time occurrences $O_i$ and $O_j$ determined by Equation (3). The Jaccard index between $x_i$ and $x_j$ is calculated by the following formula:

$$J(x_i, x_j) = \frac{|O_i \cap O_j|}{|O_i \cup O_j|}. \quad (4)$$

More precisely, the Jaccard index will quantify the frequency at which the occurrence of two events appears at an identical time. These are two identical events thanks to Definition 2 where an occurrence of two different events $x_i$ and $x_j$ in a temporal sequence $S$ must be at two different time instants. This index is on a scale from 0 to 1: 0 meaning no occurrences are identical; and 1 meaning all of them are identical.

**Example 9.** Let $x_1 = (a,t_1)$ and $x_3 = (b,t_3)$ be two events with $O_1 = \{1,18\}$ and $O_3 = \{1,18\}$. Since their time occurrences are identical, their Jaccard index is then $J(x_1,x_3) = \frac{|O_1 \cap O_3|}{|O_1 \cup O_3|} = \frac{2}{2} = 1$.

**Remark.** The restriction to have a Jaccard index equals to 1 is strong. However, there are some problems in relaxing this constraint and is the subject of on-going works. Naturally, for $S(x_i,x_j)$ to be equals to 1, $O_1$ and $O_3$ must be of the same size. Therefore, only chronicles of the same frequency can be combined.

**Algorithm 3. (Chronicle synthesis).**

1. INPUT: all 2-length chronicles
2. OUTPUT: chronicles $C_f$
3. GET maximal frequency $f_{max}$ from the 2-length chronicles
4. INIT frequency $f$ at maximal frequency $f_{max}$
5. REPEAT
6. INIT $C_f$ at empty
7. FOR all 2-length chronicles $C$ of frequency $f$
8. MERGE $C_f$ and $C$ by similarity
9. ENDFOREACH
10. DECREMENT frequency $f$
11. UNTIL all 2-length chronicles have been treated

Algorithm 3 represents the synthesis of the 2-length chronicles of the same frequency generated by step 2. First, the algorithm finds the maximal frequency $f_{max}$ of the 2-length chronicles. Next, a chronicle $C_{f_{max}}$ is created with the merging of all the 2-length chronicles of frequency $f_{max}$. This merging step is explained by Algorithm 4 given below. Then, operation is repeated for frequency $f_{max} - i$ until all the 2-length chronicles generated were processed.
Algorithm 4. (Merge by similarity operation).

1. INPUT: chronicles to merge $C_1$ and $C_2$
2. OUTPUT: merged chronicle $C_{res}$
3. INIT: chronicle $C_{res}$ with chronicles $C_1$ and $C_2$
4. FOR all events $x_i$ of events set $X_1$
5. FOR all events $x_j$ of events set $X_2$
6. IF $x_i$ and $x_j$ are similar events THEN
7. UPDATE chronicle $C_{res}$ with $x_i$ equal to $x_j$
8. ENDIF
9. ENDFOR
10. ENDFOR

Merge by similarity is presented in Algorithm 4 with this renaming operation: $C_f$ becomes $C_1$ and $C$ becomes $C_2$. First, $C_1 = (X_1, T_1)$ and $C_2 = (X_2, T_2)$ are combined in chronicle $C_{res} = (X_{res}, T_{res})$ with $X_{res} = \{X_1, X_2\}$ and $T_{res} = \{T_1, T_2\}$. Then, the Jaccard index of each events $x_i \in X_1$ and $x_j \in X_2$ are calculated with Equation (4). When $S(x_1, x_2) = 1$, events $x_1$ and $x_2$ are considered identical. In this case, $C_{res}$ is updated, $x_j$ is removed from $X_{res}$ and temporal constraints on $x_j$ are now on $x_i$. This step is repeated for all events found similar.

**Proposition 3.** The merge by similarity of two chronicles of the same frequency generated by the 2-length chronicle discovery algorithm is a chronicle.

**Proof.** To prove that $C_{res}$ is a chronicle, it is needed to prove that $X_{res}$ is a partially ordered set of events. In other words, is it possible that the operation $x_i = x_j$ does not produce a partially ordered set of events? Let $\mathcal{C} = (\mathcal{X}, T)$ with $\mathcal{X} = \{x_1 \in \{e, t_1\}, x_2 \in \{f, t_2\}\}$, $T = \{\tau_{12}\}$, and $\mathcal{C}' = (\mathcal{X}', T')$ with $\mathcal{X}' = \{x_3 \in \{e, t_3\}, x_4 \in \{e, t_4\}\}$, $T' = \{\tau_{34}\}$ be two 2-length chronicles of the same frequency. Let $S$ be a temporal sequence such that $\mathcal{I}_c(S) = \{(e, \delta_1), (f, \delta_2)\}$ and $\mathcal{I}_c(S) = \{(f, \delta_3), (e, \delta_4)\}$ are instances of $\mathcal{C}$ and $\mathcal{C}'$. With temporal constraints $\tau_{12}$ and $\tau_{34}$, inequality equations $\delta_1 < \delta_2$ and $\delta_3 < \delta_4$ are known. Let $x_1$ and $x_2$, as well as $x_3$ and $x_4$, be identical, therefore $\delta_1 = \delta_4$ and $\delta_2 = \delta_3$. However, this implies that $\delta_1$ is both strictly superior and strictly inferior to $\delta_2$. As a consequence, either $x_1$ and $x_4$, or $x_2$ and $x_3$ are different. $X_{res}$ is a partially ordered set of events and $C_{res}$ is a chronicle.

**Proposition 4.** The chronicle resulting from the merge by similarity of two chronicles of the same frequency generated by the 2-length chronicle discovery algorithm is coherent.

**Proof.** Let $C_1$ and $C_2$ be two 2-length chronicles and $C_{res}$ the chronicle created by the merge by similarity of $C_1$ and $C_2$. Let $S$ be a temporal sequence. For each couple of instance $\mathcal{I}_c(S)$ and $\mathcal{I}_c(S)$, there exists an instance $\mathcal{I}_c(S)$. Therefore, the frequency of $C_{res}$ is identical to the frequency of $C_1$ and $C_2$. As a consequence, the chronicle $C_{res}$ is coherent.

**Example 10.** Let $C_1 = (X_1, T_1)$ where $X_1 = \{x_1 = (e, t_1), x_2 = (f, t_2)\}$, $T_1 = \{\tau_{12} = x_1[255, 261]x_2\}$, $C_2 = (X_2, T_2)$ where $X_2 = \{x_3 = (e, t_1), x_4 = (e, t_2)\}$, $T_2 = \{\tau_{12} = x_3[98, 104]x_4\}$, and $C_3 = (X_3, T_3)$ where $X_3 = \{x_1 = (e, t_1), x_2 = (e, t_2)\}$, $T_3 = \{\tau_{12} = x_3[353, 365]x_2\}$ be three 2-length chronicles. Their frequencies are $f_{c_1}(S) = 4$, $f_{c_2}(S) = 4$, and $f_{c_3}(S) = 6$. After finding the maximal frequency, in this case $f_{max} = f_{c_3}(S) = 6$, all the chronicles of frequency equals to 6 are merged. Only $C_3$ is of the required frequency so $C_{res} = C_3$. Repeating this operation with $f_{max} = 5$ does not find any chronicle of this given frequency. Two chronicles of frequency equals to 4 are found, $C_1$ and $C_2$.

As seen in Figure 6, the first step is to merge them by similarity in $C_{res} = (X_{res}, T_{res})$ where $X_{res} = \{x_i = (e, t_1), x_2 = (f, t_2), x_3 = (f, t_3)\}$, and $T_{res} = \{\tau_{12} = x_1[255, 261]x_2, \tau_{13} = x_3[98, 104]x_4\}$. The Jaccard index between each event is calculated: events $x_2$ and $x_3$ are found similar. They are considered identical and the chronicle $C_{res}$ is updated such that $X_{res} = \{x_1 = (e, t_1), x_2 = (f, t_2), x_3 = (e, t_3)\}$, and $T_{res} = \{\tau_{12} = x_1[255, 261]x_2, \tau_{23} = x_2[98, 104]x_3\}$.

**Fig. 6.** Merge by similarity of two chronicles $C_1$ and $C_2$ in one chronicle $C_{res}$. The Jaccard index shows that $x_2$ and $x_3$ are similar and $C_{res}$ is created.

From the three 2-length chronicles $C_1$, $C_2$, and $C_3$ defined previously, two chronicles of different frequencies are generated: $C_{f=6} = (X_{f=6}, T_{f=6})$ where $X_{f=6} = \{x_1 = (e, t_1), x_2 = (e, t_2)\}$, $T_{f=6} = \{\tau_{12} = x_1[353, 365]x_2\}$, and $C_{f=4} = (X_{f=4}, T_{f=4})$ where $X_{f=4} = \{x_1 = (e, t_1), x_2 = (f, t_2), x_3 = (e, t_3)\}$, $T_{f=4} = \{\tau_{12} = x_1[255, 261]x_2, \tau_{23} = x_2[98, 104]x_3\}$.

**Remark.** When all the 2-length chronicles found in the previous step seen in Section 3.2 have different frequencies, this step is not required. Only chronicles of the same frequency can be combined.

### 3.4 Algorithmic complexity

In this section, the algorithmic complexity of the proposed algorithm is analyzed. It is shown that with the clustering algorithms used, a polynomial complexity can be done.

Let $n$ be the number of samples in the raw data $R$. The algorithmic complexity of LAMDA is $O(n)$, the generation of a temporal sequence from classes is also $O(n)$ complex.

The algorithmic complexity of the temporal sequence extraction step is $O(n)$.

Let $\frac{\binom{1}{1}}{2}$ be the number of temporal distances found in the l-length temporal sequence $S$ input in the 2-length chronicle discovery. The number of discovered 2-length chronicles is defined by $c_2 = \frac{\binom{1}{1}}{2}$. The inputs of
Fig. 7. Normalized values of the first 1000 samples of the features Pitch and Yaw of dataset \( \mathcal{R} \). \( \mathcal{R} \) is a record of the right arm movement of a professional athlete in a swimming scenario at a sampling rate of 50\(Hz\).

Fig. 8. Classes found by LAMDA on the first 1000 samples of dataset \( \mathcal{R} \). The under class represent the arm pulling and pushing underwater, whereas in the above class, the arm is recovering above the water.

DBSCAN is \(c_2\), as such, its complexity is \(O(c_2 \log(c_2))\) that can be simplified by \(O(l^2 \log(l^2))\). So, the algorithmic complexity of the 2-length chronicle discovery step is \(O(l^2 \log(l^2))\) and depends on the complexity of the clustering algorithm used.

The algorithmic complexity of the merge by similarity operation between a \(m\)-length chronicle \(C_1\) and a 2-length chronicle \(C_2\) is given by \(O(m \log(m))\). The chronicle synthesis step will process at most \(c_2\) merge by similarity operations. So, the complexity of this step is \(O(c_2 m \log(m))\).

The overall algorithmic complexity of the chronicle discovery from raw data algorithm is given by:

\[
O(n^2 m \log(m)),
\]

with \(n\) the number of samples in the raw data \(\mathcal{R}\) and \(m\) the length of the longest discovered chronicle. The complexity is highly dependent of the choice of the clustering algorithm parameters implemented in step 2. Badly chosen parameters can produce a high number of 2-length chronicles \(c_2\). This problem shows the necessity to enforce a quality check at the end of the 2-length chronicle discovery step to limit the impact of \(c_2\) on the discovery time.

4. APPLICATION

In this section, an application about health monitoring of an athlete movement is detailed. More precisely, data from an instrumented glove for swimming performance monitoring (Mangin et al., 2015) are captured when the athlete performs a front crawl on a swimming pool.

The device consists in several sensors (accelerometers, magnetometers, gyroscopes) that allow a precise description of the device movements with a sampling rate of 50\(Hz\). This equipment is worn on the right hand in order to record the right arm movements in a swimming situation. The dataset \(\mathcal{R}\) is a recording of an athlete performing a front crawl on a long course swimming pool (50\(m\)).

The time-ordered dataset \(\mathcal{R}\) contains 2051 samples with 16 features. The normalized features in the [0, 1] interval are divided as follows: Euler angles Roll, Pitch, and Yaw; quaternions \(Q_1\), \(Q_2\), \(Q_3\), and \(Q_4\); accelerometers \(Acc_x\), \(Acc_y\), and \(Acc_z\); gyroscopes \(Gyro_x\), \(Gyro_y\), and \(Gyro_z\); and magnetometers \(Mag_x\), \(Mag_y\), and \(Mag_z\). Figure 7 gives the first 1000 samples of the Euler angle features where we can see already a repetition of a pattern.

The temporal sequence extraction step is performed first on the 16 features of \(\mathcal{R}\). The LAMDA algorithm is used with the fuzzy centered binomial method to calculate the MADs, the probabilistic method to calculate the GADs, and the exigency level \(\alpha\) sets to 1 and \(\delta\) sets to 5. With those parameters, LAMDA finds two classes in \(\mathcal{R}\). These classes are interpreted as follow: in the first class, denoted under, the arm is pulling and pushing under the water; in the second class, denoted above, the arm is recovering above the water. Figure 8 gives the class of the first 1000 samples of the dataset \(\mathcal{R}\).

Once the temporal sequence is extracted, the next step is the 2-length chronicle discovery. With the DBSCAN parameter \(minPts\) set to 1 and \(\varepsilon\) set to 12, a total of 62 2-length chronicles are discovered with their frequencies ranging from \(f_{min} = 2\) to \(f_{max} = 17\).

Finally, the chronicle synthesis step is processed on all the 62 2-length chronicles. One of the chronicle generated by this step called \(C_{res} = (X_{res}, T_{res})\) with \(X_{res} = \{x_1 = (above, t_1), x_2 = (under, t_2), x_3 = (above, t_3)\}\), and \(T_{res} = \{t_{12} = x_1[48, 69], t_{13} = x_1[106, 130], t_{23} = x_2[52, 62], t_{31} = x_3[106, 130]\}\). This chronicle has a frequency of \(f_{max} = 17\).

Physically, this chronicle represents a complete arm movement, called stroke cycle. Taking into account the sampling rate of 50\(Hz\) of the raw data, one could discriminate several informations: the recovering phase is done in the interval of [0.96, 1.38] seconds, represented by the temporal constraint \(\tau_{12}\); the underwater phase is done in the interval of [1.04, 1.24] seconds, represented by the temporal constraint \(\tau_{23}\); the complete stroke cycle is done in the interval of [2.12, 2.6] seconds, represented by the temporal
The discovered chronicle describes the nominal arm movements of the athlete performing a front crawl and will be recognized if no discrepancy between the optimal movements of the athlete and the movements recorded in another dataset exist. Fault diagnosis of movement deviations in the temporal domain, in the event domain, or both, considered as faults for the athlete can be done when the chronicle is not recognized when it should be.

5. CONCLUSION

This paper provides a clustering approach for designing chronicles with minimal knowledge from the dynamic process to diagnose. The algorithm presented uses a clustering method based on the fuzzy logic to construct a temporal sequence from raw data. Chronicles of different frequencies are then learned from this temporal sequence as a result of a density based clustering algorithm. The chronicle discovery algorithm proposed in this paper is done with a polynomial algorithmic complexity. An application of real data from a swimming performance monitoring device for health monitoring of the athlete movements is detailed.

Further works need to be done to generalize the designed chronicles. Presently, a limitation lies in the fact that only one temporal sequence $S$ is constructed, as a consequence, there is no guarantee that designed chronicles can be recognized in another temporal sequence $S'$ with a slightly different behavior. Another idea is to exploit the Jaccard index not only on a crisp value (0 or 1) but on a fuzzy value (a percentage of similarity). This could permit to combine chronicles with different frequencies. Finally, it could be interesting to design a chronicle discovery algorithm that could deal with domain constraints, when an event could not occur in a period of time; and event counters (added in the chronicle representation in (Dousson, 2002)), when a determined number of events must occur in a given time interval.

ACKNOWLEDGEMENTS

We are very grateful to Aurélien Valade, Pascal Acco, Georges Soto-Romero, and the members of the Smart Sensing and Systems Monitoring (S4M) team of the LAAS-CNRS for allowing us to use their data presented in Section 4 of this paper.

REFERENCES


