TLV-diss

γ

: A Dissimilarity Measure for Public Administration Process Logs

Flavio Corradini, Caterina Luciani, Andrea Morichetta, Marco Piangerelli, Andrea Polini

TLV-diss$_\gamma$: A Dissimilarity Measure for Public Administration Process Logs

Flavio Corradini, Caterina Luciani ♠, Andrea Morichetta, Marco Piangerelli, and Andrea Polini
University of Camerino, Camerino (MC) 62030, IT
name.surname@unicam.it

Abstract. Every day Public Administrations (PA) provide citizens with plenty of services. Due to different factors, such as the involvement of different human resources or the will to deliver lean and versatile services, the same service can show some variability across different organizations. Log files contain the proof of PA process' variability thus, being able to analyze logs, can be very helpful both for the PA, in order to establish good practices or contextual rules, as well as for the software house companies that need to analyse and to better customize the software they provide. In this paper, we present a methodology that, using log files as inputs, and based on the so-called TLV-diss$_\gamma$, a parametric dissimilarity measure, allows a data analyst to perform a cluster analysis. This methodology helps both PA and software producers to better understand how services are delivered through informative systems and then to better customize them. We show that our methodology can be used to capture the differences in control flow and components resulting from the log files, and then to better reason on the delivery of public services.

Keywords: Log clustering · Similarity measure · Variance Analysis

1 Introduction

Every day PA provide citizens with plenty of services that are very similar in scope, but that may vary in their internal management and organisation. Such divergence in the processes is called variability and can be due to many possible factors, such as differences in the human resources involved to carry out specific tasks, altered control flow necessary to deal with specific locally-applicable laws or requirements, specific aspects of external informative systems with which the supporting software has to interact, and to many other factors that could require specific customization on the delivery of public service.

Inevitably, this variability increases the complexity of data analysis in discovering possible deviations from expected procedures in which the PA analyst could be interested. In parallel, a software company, distributing services and applications for PA, could improve and optimize software, reducing the investment in the development for uncommon behaviors and, at the same time, tuning the software for including non-standard procedures that are, instead, very common among PA employees. Moreover, given different installations of heterogeneous organisation, identifying regularities in activities or control flow, that may reveal good practices or contextual rules in the execution of a standard procedure, is very important for predictive installation configurations. In such a context, the

data analyst has a unique source for discovering possible anomalies and elaborate possible statistics: the big amount of raw data collected in log files, generated from the execution of the software, supporting the delivery of the public service. Process mining is a recently developed discipline that aims at discovering business process models from log files in order to enable more abstract reasoning on the behaviour subsumed by a service and to possibly compare different service instances in relation to the behavioural characteristics associated to the derived business process models. Notwithstanding its numerous merit, the main limitation of this approach concerns the limited number of logs that can be analyzed concurrently, when, instead, thousands of comparisons would be needed. In particular, if we would like to carry on extensive analysis of the logs related to the delivery of a specific service by the municipalities of an average size European country, we would need to compare several thousand models. At the same time, the result could be strongly influenced by the adopted mining algorithm.

The methodology we propose is propaedeutic to Process Mining and it tries to solve some well-known problems when carrying on extensive log files comparison and clustering. Our aim is to satisfy the needs of PA data analysts, for instance, employed within a large PA software provider, equipping them with an instrument able to help them in collecting and comparing information derived from a huge amount of logs, generated by different municipalities. The methodology is based on a new (dis)similarity measure, named TLV-diss$_\gamma$, that is able to quantify on a numerical scale the pairwise (dis)similarity between logs, starting from their XES files. This measure will be used successively in a clustering procedure in order to group together comparable logs thus enabling a faster and cluster-based analysis. For such a reason, the proposed methodology is parametric and can be configured directly by the data analyst. According to the parameters' value, the similarity measure will give relevance to different characteristics of a log. In particular, the analyst will be able to focus more on the activity perspective (i.e. on how much logs are different in terms of reported performed activities) than on a control perspective instead (i.e. on how much logs report different ordering relations among the activities). Moreover, it is possible to have a trade-off between the two. Our methodology has been implemented in a tool and validated in relation to its effectiveness on 37 logs related to the same service and delivered by different Italian municipalities.

The rest of the paper is organized as follows: Section 2 presents the background, in Section 3 we provide the theoretical and technical details about the methodology that was applied to a real case study, and whose results are presented in Section 4. In Section 5 we have the related works and finally, in Section 6 we conclude and discuss possible future works.

2 Background

2.1 Information Systems and Event Logs

Public Administrations deliver complex services to citizens through the usage of dedicated Information Systems (IS). In the last years, the development of such systems have been more and more driven by the definition of business processes

that are then embedded in the software [3], so that the activities to be performed in the delivery of a service and their ordering, are clearly specified and checked with respect to the norm. In such a context the term Process Aware Information System (PAIS) has been defined to refer to such kind of IS.

Among the various functionality made available, a PAIS generally includes diagnostic mechanisms in the form of events log. An event log is a collection of data connected to the execution of the embedded process, and each event in the log is associated with a specific process instance, which is usually referred to as a "case" [18]. A "case" is then constituted by an ordered sequence of events that are related to the delivery of a service in relation to a specific request by a citizen. There can be additional attributes for each event, such as the timestamp or the resource executing the activity. Each attribute may be considered as a classifier. If two events have the same value for a classifier they are considered equal. The default classifier is the name of the activity. Since 2010, the XES format has been the most widely used format in process mining to analyse event logs [18]. An XES document depicts a single log and an arbitrary number of traces. Each trace describes a sequence of events attributable to a single case. The log can have as many attributes as needed, which are mainly of type String, Date, Int, Float, and Boolean.

2.2 Clustering and K-medoids Algorithm

Clustering analysis is a set of techniques for grouping (clustering) similar objects together, so that the dissimilarity among objects belonging to the same cluster is lower than the one among objects belonging to different clusters [6].

Among partitional clustering algorithms, K-means and K-medoids are very popular. They are based on a simple concept: be K, the number of clusters, the algorithms look for the K-most-representative elements, called centroids or medoids, and assign each object to its closest representative. One of the most relevant differences between the two algorithms is given by the selection of the medoids that are chosen among the collected data samples, while the centroids can be any point in the considered space [9]. It is for this reason that k-means is not suitable for non-Euclidean spaces (as in our case), and k-medoids has been used instead [17]. For optimizing the number of clusters K, some performance indexes, such as the silhouette index, have been defined [15]. Basically, varying K we look for the partition that ensures the maximum silhouette index. The method consists of three steps: calculate the average distance of a point from points in the same cluster $a(i)$, count the average distance of the same point from points in the nearest cluster $b(i)$, calculate $\frac{b(i)-a(i)}{max(b(i),a(i))}$.

3 TLV-diss$_\gamma$ Methodology

In this section, we present the TLV-diss$_\gamma$ methodology. It starts with the elaboration of the XES logs to successively make possible the application of the K-medoid algorithm on a generated distance matrix.

The proposed methodology consists of five phases executed in sequence:

– Definition of the Trace Matrices for each XES log

– Definition of the Log Matrix combining together the Trace Matrices
– Definition of the Variance Matrix for each possible combination of logs
– Definition of the Distance Matrix summarizing the results of the Variance Matrices
– Use of K-medoid algorithm on logs according to the Distance Matrix

In such a context, the term trace can be considered as the formal representation of a case, and it will permit to formally represent the sequence of activities performed to satisfy a request by a citizen.

3.1 Trace Matrix

A Trace Matrix defines the ordering relationship between activities in a trace extracted from a XES Log. In particular, a Trace Matrix is generated for each distinct trace existing in the Log. In this definition, we aim to extend the concept of order matrix already introduced by Reichert et al. [11].

Definition 1. *Let $\mathcal{A}$ a generic set of labels, denoting activities, being $a \in \mathcal{A}$ a generic activity. A* ***trace*** *σ is represented as an ordered sequence of activities, so that $\sigma = \langle a_1, a_2, ..., a_r \rangle$. We indicate with $\mathcal{A}^\sigma$ the set of labels appearing in the trace σ. A* ***log*** *$\mathcal{L}$ is constituted by a set of traces, so that $\sigma \in \mathcal{L}$. The set of all labels appearing in any trace of a log is then represented by $\mathcal{A}^\mathcal{L}$ (i.e. $\mathcal{A}^\mathcal{L} = \bigcup_{i=1}^{n} \mathcal{A}^{\sigma_i}$ where $\mathcal{L} = \{\sigma_i | i \in [1 \dots n]\}$). We indicate with $|\mathcal{A}^\mathcal{L}|$ the cardinality of a set.*

Definition 2. *Given a trace σ the associated* ***trace matrix*** *$\mathcal{T}^\sigma$ will report the relations among the activities in the trace σ itself. $\mathcal{T}^\sigma$ is a squared matrix that has a number of rows, and columns, equal to $|\mathcal{A}^\mathcal{L}|$. Each row is associated to a label in $\mathcal{A}^\mathcal{L}$, and the same label is associated to the column with the same index. Therefore, the elements of $\mathcal{T}^\sigma$ are defined as follows:*

– $t^\sigma_{ij} := \triangleright$ *if a_i precedes, directly, a_j in σ*
– $t^\sigma_{ij} := \not\triangleright$ *if a_i precedes, but not directly, a_j in σ*
– $t^\sigma_{ij} := \triangleleft$ *if a_i follows, directly, a_j in σ*
– $t^\sigma_{ij} := \not\triangleleft$ *if a_i follows, but not directly, a_j in σ*
– $t^\sigma_{ij} := \emptyset$ *if a_i and a_j do not both appear in σ*

Given a log $\mathcal{L}$ the set of all trace matrices corresponding to the traces in $\mathcal{L}$ will be denoted as $\mathcal{T}^\mathcal{L}$.

It is worth clarifying that an activity a_i precedes/follows directly a_j if it is in its immediate adjacency, so respectively it holds that $j = i + 1$ or $i = j + 1$; not directly means that one activity can precede/succeed the other in more than one step. For each log, the activities considered in the trace matrix corresponds to the activities existing in that specific log and not only the trace. This is necessary to have a more direct generation of the log matrix in the next step. Moreover, it is worth noticing that the expressed relations do not hold on to the same activity ($i = j$, $\forall i, j$). For this reason the diagonal is empty, and T^σ is "specular" with respect to its diagonal. Finally, in the proposed definition we treat equally activities that are reachable in more than two steps, differently from [8]. A more detailed distinction is possible, but it could affect the computational performance in dealing with a huge amount of data.

Running Example (1/4) To better illustrate the technicalities of our methodology we present here a running example. Let $\mathcal{L}_1$ be the log represented in Table 1a and $|\mathcal{L}_1| = n$ (i.e. $\mathcal{L}_1$ includes n traces). Then let the $\mathcal{L}_2$ be the log depicted in Table 1b.

Case ID	**Activity name**
1	A
1	B
1	D
2	A
2	C
2	D
...	...

(a) $\mathcal{L}_1$

Case ID	**Activity name**
1	A
1	B
1	C
1	E
2	A
2	C
2	B
2	E
...	...

(b) $\mathcal{L}_2$

Table 1: Running example logs

	A	B	C	D
A		$\triangleright$	$\emptyset$	$\not\triangleright$
B	$\triangleleft$		$\emptyset$	$\triangleright$
C	$\emptyset$	$\emptyset$		$\emptyset$
D	$\not\triangleleft$	$\triangleleft$	$\emptyset$	

(a) ABD $\mathcal{L}_1$

	A	B	C	D
A		$\emptyset$	$\triangleright$	$\not\triangleright$
B	$\emptyset$		$\emptyset$	$\emptyset$
C	$\triangleleft$	$\emptyset$		$\triangleright$
D	$\not\triangleleft$	$\emptyset$	$\triangleleft$	

(b) ACD $\mathcal{L}_1$

	A	B	C	E
A		$\triangleright$	$\not\triangleright$	$\not\triangleright$
B	$\triangleleft$		$\triangleright$	$\not\triangleright$
C	$\not\triangleleft$	$\triangleleft$		$\triangleright$
E	$\not\triangleleft$	$\not\triangleleft$	$\triangleleft$	

(c) ABCE $\mathcal{L}_2$

	A	B	C	E
A		$\not\triangleright$	$\triangleright$	$\not\triangleright$
B	$\not\triangleleft$		$\triangleleft$	$\triangleright$
C	$\triangleleft$	$\triangleright$		$\not\triangleright$
E	$\not\triangleleft$	$\triangleleft$	$\not\triangleleft$	

(d) ACBE $\mathcal{L}_2$

Table 2: Trace Matrices

The resulting trace matrices for $\mathcal{L}_1$ are Table 2a for the ABD trace and Table 2b for the ACD trace. Similarly, for $\mathcal{L}_2$ we have Table 2c and Table 2d for the traces ABCE and ACBE respectively.

3.2 Log Matrix

A log matrix summarizes the behavior reported in the log and is obtained by the trace matrices previously generated. In addition to the relationships already described in the trace matrices, here we can express the relation of XOR or AND between activities.

In particular we use:

- $\times$, if the two activities never appear on the same trace (XOR);
- $+$, if there is at least one trace in which the activity t_i appears before the activity t_j and another trace with the same activities but with the inverse order (AND).

A log matrix is obtained by comparing all trace matrices of the Log.

Definition 3. *Let $\mathcal{LM}$ be the **Log Matrix** corresponding to a log $\mathcal{L}$, then a generic entry in $\mathcal{LM}$ is indicated as l_{ij} and is defined as follows (taking into account the trace matrices corresponding to the traces in log $\mathcal{L}$) :*

- $l_{ij} := \triangleright$ *if* $\exists \sigma \in \mathcal{L} : t^{\sigma}_{ij} = \triangleright$ *and* $\forall \sigma' \in (\mathcal{L} \setminus \sigma)$*:* $t^{\sigma'}_{ij} = (\triangleright$ *or* $\emptyset)$
- $l_{ij} := \triangleleft$ *if* $\exists \sigma \in \mathcal{L} : t^{\sigma}_{ij} = \triangleleft$ *and* $\forall \sigma' \in (\mathcal{L} \setminus \sigma) : t^{\sigma'}_{ij} = (\triangleleft$ *or* $\emptyset)$
- $l_{ij} := \not\triangleright$ *if* $\exists \sigma \in \mathcal{L} : t^{\sigma}_{ij} = \not\triangleright$ *and* $\forall \sigma' \in (\mathcal{L} \setminus \sigma) : t^{\sigma'}_{ij} = (\triangleright$ *or* $\emptyset)$
- $l_{ij} := \not\triangleleft$ *if* $\exists \sigma \in \mathcal{L} : t^{\sigma}_{ij} = \not\triangleleft$ *and* $\forall \sigma' \in (\mathcal{L} \setminus \sigma) : t^{\sigma'}_{ij} = (\triangleright$ *or* $\emptyset)$
- $l_{ij} := \times$ *if* $\forall \sigma \in \mathcal{L} : t^{\sigma}_{ij} = \emptyset$*;*
- $l_{ij} := +$ *if* $\exists \sigma \in \mathcal{L} : t^{\sigma}_{ij} = (\not\triangleright$ *or* $\triangleright)$ *and* $\exists \sigma' \in (\mathcal{L} \setminus \sigma) : t^{\sigma'}_{ij} = (\not\triangleleft$ *or* $\triangleleft)$

Running Example (2/4) Considering the Trace Matrices presented in Table 2, here below we define the corresponding Log Matrices following the rules defined above. In particular Table 3a shows the log matrix for $\mathcal{L}_1$ merging together the trace matrices defined in Table 2a and 2b. Table 3b represents the log matrix for $\mathcal{L}_2$ regarding the trace matrices defined in Table 2c and Table 2d.

	A	B	C	D
A		$\triangleright$	$\triangleright$	$\not\triangleright$
B	$\triangleleft$		$\times$	$\triangleright$
C	$\triangleleft$	$\times$		$\triangleright$
D	$\not\triangleleft$	$\triangleleft$	$\triangleleft$	

(a) $\mathcal{L}_1$

	A	B	C	E
A		$\not\triangleright$	$\not\triangleright$	$\not\triangleright$
B	$\not\triangleleft$		$+$	$\not\triangleright$
C	$\not\triangleleft$	$+$		$\not\triangleright$
E	$\not\triangleleft$	$\not\triangleleft$	$\not\triangleleft$	

(b) $\mathcal{L}_2$

Table 3: Log Matrices

3.3 Variance Matrix

A Variance Matrix defines the distance between two log files. Each entry of the Variance Matrix is obtained by comparing the corresponding element of two Log Matrices and by weighting the result using parameters γ and β.

Definition 4. *Let $\mathcal{V}$ be a* ***Variance Matrix*** *corresponding to the logs $\mathcal{L}$ and $\mathcal{L}'$ then the elements of such a matrix are defined as follows:*

$$v_{ii} = \begin{cases} \gamma, & \text{if } a_i \in \mathcal{A}^{\mathcal{L}} \cap \mathcal{A}^{\mathcal{L}'} \\ 0, & \text{otherwise} \end{cases}$$

$$v_{ij} = \begin{cases} (1-\gamma), & \text{if } l_{ij} \text{ matches } l'_{ij} \\ \beta \cdot (1-\gamma), & \text{if } l_{ij} \text{ and } l'_{ij} \text{ have same order, with } \beta < (1-\gamma) \\ 0, & \text{otherwise} \end{cases}$$

where l_{ij} and l'_{ij} are the elements of the Log Matrices corresponding to logs $\mathcal{L}$ and $\mathcal{L}'$, respectively.

In particular, the entries corresponding to the diagonal of the matrix are used to keep track of the presence of the activities in the logs, while the remaining part of the matrix is used to annotate the relationship between them. If the activity a_i is present in the intersection of $\mathcal{A}^{\mathcal{L}}$ and $\mathcal{A}^{\mathcal{L}'}$ then the corresponding entries in the diagonal are filled with γ, 0 otherwise.

Considering instead the entries v_{ij} they are filled with $(1-\gamma)$ if the entries of the Log Matrices for $\mathcal{L}$ and $\mathcal{L}'$ are identical; $\beta \cdot (1-\gamma)$ if they respect the same order $(\lhd, \not\lhd)$ or $(\rhd, \not\rhd)$, 0 otherwise.

The parameter γ is used to control the impact of activities with respect to the relationship between them. Thus, it can be understood that it is possible to give more weight to the common existence of the activities giving more weight to the elements on the diagonal (γ close to 1) or more weight to the relationships between activities (flow) giving more weight to the elements that are not on the diagonal (γ close to 0). The parameter β is useful for discriminating that cases where the entries l_{ij} and l'_{ij} are not identical but still respect the same order of precedence. The usage of these parameters permits the final user to customize the measure according to his/her necessity and interest.

To notice, in order to compare matrices with different activities we use in the Variance Matrix the super-set of the activities between the two Log Matrices ($\mathcal{A}^{\mathcal{L}} \cup \mathcal{A}^{\mathcal{L}'}$) and we insert 0 on the entries that refer to the absent activity.

Running Example 3/5 Considering the two Log Matrices defined in Table 3a and 3b the resulting Variance matrix considering the parameter γ=0.6 and the parameter β=0.25 is defined in Table 4.

	A	B	C	D	E
A	0.6	0.1	0.1	0	0
B	0.1	0.6	0	0	0
C	0.1	0	0.6	0	0
D	0	0	0	0	0
E	0	0	0	0	0

Table 4: Variance Matrix $\mathcal{V}$ corresponding to logs $\mathcal{L}_1$ and $\mathcal{L}_2$

3.4 Distance Matrix

The Distance Matrix summarizes the similarity between Logs using a range between 0 and 100. In particular, the closer the value is to zero, the more similar the two models are.

Definition 5. *Given a set of logs let $\hat{V}$ be the set of all Variance Matrices. Then $\mathcal{D}$ is the **Distance Matrix** among Variance Matrices, where each element of the matrix $d_{ij} = diss(\mathcal{L}_i, \mathcal{L}_j)$, is computed using equation (3)*

Given $|\hat{\mathrm{V}}| = N$, $\mathcal{D}$ is a matrix of order N.

It is worth clarifying that a Variance Matrix $\mathcal{V}$ contains the similarity among the two Log Matrices associated with the logs $\mathcal{L}$ and $\mathcal{L}'$. In order to derive a value (real number), for quantifying the similarity between two Logs, we used the following relationships:

$$\mathrm{sim}(\mathcal{L}_i, \mathcal{L}_j) = \frac{1}{\alpha}||\mathcal{V}||_1 = \frac{1}{\alpha}\sum_{i}^{N}\sum_{j}^{N}|v_{ij}| \tag{1}$$

$$\alpha = \begin{cases} n, & \text{if } \gamma = 1 \\ n(n-1), & \text{if } \gamma = 0 \\ n^2, & \text{otherwise} \end{cases} \quad (2)$$

$$\text{diss}(\mathcal{L}_i, \mathcal{L}_j) = (1 - \text{sim}(\mathcal{L}_i, \mathcal{L}_j)) * 100 \quad (3)$$

The similarity $\text{sim}(\mathcal{L}_i, \mathcal{L}_j)$ is calculated by dividing per α the 1-norm of the corresponding Variance Matrix $\mathcal{V}$, where α assumes a different value according to the definition (2). In other words, α should assume a value equal to the number of cells possibly different from 0 according to the definition of the parameter γ. Finally, the dissimilarity of two logs, $\text{diss}(\mathcal{L}_i, \mathcal{L}_j)$, is defined as one minus the similarity, expressed in hundredths.

Running Example(4/4) The distance matrix (Table 5) of the considered running example is defined on the dissimilarity of logs calculated in Table 4. In particular, for logs $\mathcal{L}_1$ and $\mathcal{L}_2$, we have the following distance:
$\text{diss}(\mathcal{L}_1, \mathcal{L}_2) = (1 - \text{sim}(\mathcal{L}_1, \mathcal{L}_2)) * 100 = (1 - (\frac{(0.6*3)+(0.1*4)}{25})) * 100 = 91.2$

	Log_1	Log_2
Log_1		91.2
Log_2		

Table 5: Distance Matrix

3.5 Logs Clustering

The last step of our methodology consists in defining the Log clusters according to the generated Distance Matrix. The clustering algorithm that we take into consideration is the K-medoids for its characteristic to select the medoids among the collected Logs. The number of partitions K is selected automatically by the algorithm ensuring the maximum silhouette index.

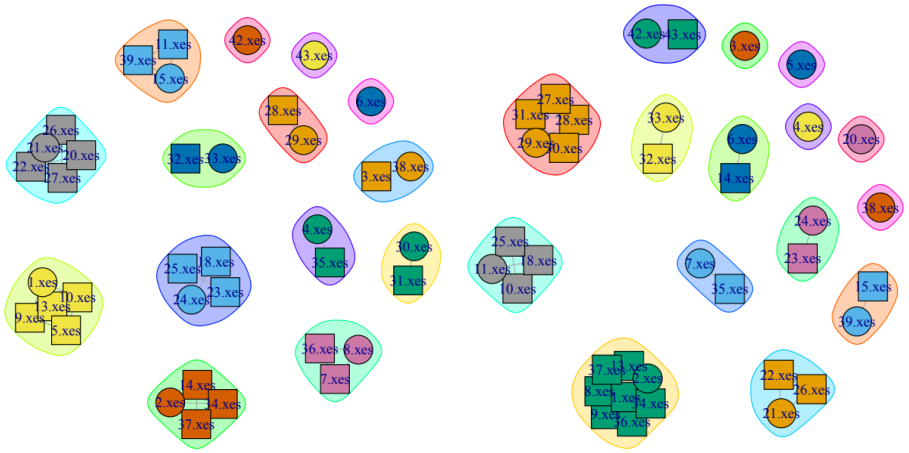
4 Empirical Results / Case Study

To study and evaluate the performances of our (dis)similarity measure, we analyzed data provided by a company that sells information systems to the PA. We selected a single service offered by municipalities and we derived a set of logs that have been successively pre-processed by a semi-automatic procedure in order to anonymize and unify activities so to run our algorithm. The pre-processing phase permitted us to select 37 logs describing the same process. The first four phases of the proposed methodology were implemented in a Java tool[1]. In particular, after the selection of a folder containing the XES Logs to analyze the tool is able to derive a csv file containing the Distance Matrix between all Logs. Successively, the generated csv file has been imported in R, for performing

[1] https://bitbucket.org/proslabteam/tlv-diss/downloads/

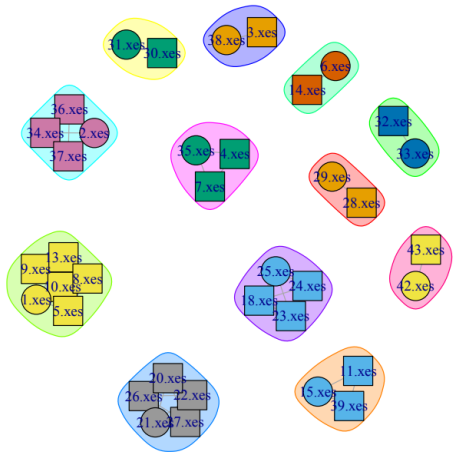
the K-medoids algorithm thanks to the *cluster* library[2]. To evaluate and test the performance of the defined (dis)similarity measure and the corresponding cluster results, three configurations were studied. In particular, we set the parameter β=0.1 and the γ varying as follows:

- $\gamma = 0$ to reward control flow similarity (C_{Flow})
- $\gamma = 1$ to reward activities similarity (C_{Act})
- $\gamma = 0.5$ to equally reward control flow and activities ($C_{Fl/Act}$)



(a) $\gamma = 0$: similar control flow

(b) $\gamma = 1$: similar activities



(c) $\gamma = 0.5$: mix of control flow and activities

Fig. 1: Clustering. Each Log file is indicated using the XES extension; the circles represents the Medoids, i.e. the cluster's most representative object

[2] https://cran.r-project.org/web/packages/cluster/cluster.pdf

The optimal number of clusters K is found by calculating the Average Silhouette Value, S_{average}, for several K. The K-medoids algorithm then will select the K at which S_{average} is maximum. This guarantee the best results of the algorithm. We obtained that K= 14 for configuration C_{Flow}, K= 15 for configuration C_{Act} and K= 12 for configuration $C_{Fl/Act}$.

Fig. 1 shows the clusters obtained for each configuration. By observing Fig. 1a and 1b, we see that Log_{42} and Log_{43} appear in different clusters if $\gamma = 0$, while they belong to the same cluster if $\gamma = 1$. This fact is somehow expected, in fact $Log_{42} = \{\langle \text{H,G} \rangle, \langle \text{T,Q} \rangle\}$, while $Log_{43} = \{\langle \text{T,G,Q,H} \rangle, \langle \text{G,T,H,Q} \rangle\}$. This shows how our methodology allows the analyst to choose whether to give more weight to the similarity of activities or of control flows.

Log_6 and Log_{14} show a very interesting behavior, too. $Log_6 = \{\langle \text{C,A} \rangle\}$, while $Log_{14} = \{\langle \text{A,C} \rangle\}$. On the one hand, choosing $\gamma = 0$, Log_6 is a cluster of its own and Log_{14} is clustered with $Log_2 = \{\langle \text{A,B,C,E} \rangle, \langle \text{A,C,B,E} \rangle\}$, $Log_{34} = \{\langle \text{A,C} \rangle, \langle \text{A,B} \rangle\}$ and $Log_{37} = \{\langle \text{B,A,C} \rangle\}$. This is not surprising, in fact in this cluster the existence of the same $\langle \text{A,C} \rangle$ sub-sequence is identified. On the other hand, as can be seen in Fig. 1a and 1c, fixing $\gamma = 1$ and $\gamma = 0.5$, Log_6 and Log_{14} are in the same cluster: this behavior is consistent because the Logs share both activities.

Finally, $Log_4 = \{\langle \text{A,B,C} \rangle\}$ and $LOG_{35} = \{\langle \text{A,B} \rangle\}$ are clustered together when $\gamma = 0$ while if $\gamma = 1$, Log_{35} is cluster with $Log_7 = \{\langle \text{B,A} \rangle\}$ and if $\gamma = 0.5$ the three Logs above belong to the same cluster as shown in Fig. 1c.

While a thoughtful assessment of the approach is certainly needed, so far the results we got in terms of performance seem to support its applicability to the comparison of many logs. In particular, ten runs on our data-set composed of 37 logs the final Distance Matrix was generated with an average time of 4.3 seconds, using a consumer PC.

5 Related Works

There is a large literature on similarity measures between business models or logs. In [16] the authors identify four dimensions to quantify the similarity between models and between their elements.

The natural language dimension provides a syntactic comparison of labels in the model to determine the equality between two labels, or the semantic one, to identify the synonyms of the labels. The graph structure is the dimension that allows comparison of e.g. the largest sub-pattern in common between the two graphs or the position and type of the various control flow connectors. The human estimation dimension allows human opinion to be included in the judgment of how similar two processes are. Finally, behaviour dimension compares trace executions. Gerke et al. [7] and Zhou et al. [19] use the longest sub-traces to express Log similarity. In [4] van Dongen et al. use causal footprints to define relationships between activities. The similarity is then calculated by using the cosine distance. Kunze et al. [10] define a metric from order relationships between trace elements, i.e. strict order, exclusiveness, interleaving, and co-occurrence. The metric provides a weighted contribution of the similarity between the two models according to each of the order relationships. Our similarity measure dif-

fers from the mentioned works because it takes all traces into account, has the ability to consider AND and XOR, and thanks to the concept of follows (precedes) strictly, it is possible to express notions about the context. The effectiveness of our similarity measure is also tested through a clustering technique making our methodology suitable for large amounts of data.

Model clustering has already been explored in the literature, although existing works take business models as input. Our work instead involves the analysis of execution logs and this methodology is very suitable in case of a large number of logs, as using process mining techniques to derive the model would be very expensive. In [8] the authors use a metric that equally weighs the contribution of activities and transitions. They use hierarchical clustering to group the models. Again, using models, Bergmann et al.[1] tested the effectiveness of k-medoid for partitioning a recipe repository, in [5] the authors use clustering to identify fragments of patterns present in several processes that can be refactored as shared processes and in [13] the authors adopt a similarity function based on fuzzy logic, which is then used to cluster the models.

6 Conclusions and Future Works

In this paper, we derived a parametric dis(similarity) measure working on log files produced by PAs. Our approach, starting from the traces describing the executions of services, derives a distance matrix representing the (dis)similarity among logs, so to permit a deeper understanding of variability in the delivery of services to citizens by different organizations. The proposed methodology is particularly suited for comparing and analyzing large amounts of data, where standard techniques (e.g. process mining) are not indicated due to scalability issues. Furthermore, our approach is able to deal with the real behavior of systems, since it is completely data-driven, and does not rely on the knowledge of any model, which in general could not precisely reflect real IS executions.

The integration of a clustering procedure in the proposed methodology is twofold: first of all, it allows to evaluate the discriminating power of the TLV-diss$_\gamma$ measure, secondly, it permits to aggregate and analyze similar logs in order to provide the analyst with a more focused and complete view on the global behavior of the group.

As shown in section 4, we obtained promising results on the analyzed logs. Indeed, the proposed parametric measure was able to discriminate logs according to the selected parameters, showing the efficacy and effectiveness of the approach. This confirms the goodness of the used parameters. Anyway, larger and deeper investigations on such field are needed, such as the possibility to introduce other perspectives in addition to control flow and activities or the possibility to test the application in other scenarios [2]; as well as using Topological Data Analysis-based clustering [12, 14]. Furthermore, possible improvements of the dis(similarity) measure are under study.

Acknowledgements Caterina Luciani's work has been funded by Maggioli Spa

References

1. Bergmann, R., Müller, G., Wittkowsky, D.: Workflow clustering using semantic similarity measures. In: Annual Conf. on AI. pp. 13–24 (2013)
2. Corradini, F., Marcantoni, F., Morichetta, A., Polini, A., Re, B., Sampaolo, M.: Enabling auditing of smart contracts through process mining. Lecture Notes in Artificial Intelligence and in Bioinformatics **11865**, 467–480 (2019)
3. Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Tiezzi, F.: Engineering trustable choreography-based systems using blockchain. In: Symposium on Applied Computing. p. 1470–1479. SAC '20, Ass. for Computing Machinery (2020)
4. van Dongen, B., Dijkman, R., Mendling, J.: Measuring similarity between business process models. In: Advanced IS Engineering. pp. 450–464 (2008)
5. Ekanayake, C.C., Dumas, M., García-Bañuelos, L., La Rosa, M., ter Hofstede, A.H.: Approximate clone detection in repositories of business process models. In: BPM. pp. 302–318. Springer (2012)
6. Friedman, J., Hastie, T., Tibshirani, R., et al.: The elements of statistical learning, vol. 1. Springer series in statistics New York (2001)
7. Gerke, K., Cardoso, J., Claus, A.: Measuring the compliance of processes with reference models. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 76–93. Springer (2009)
8. Jung, J.Y., Bae, J., Liu, L.: Hierarchical clustering of business process models. Int. J. of Innovative Computing, Information and Control **5**(12), 1349–4198 (2009)
9. Kaufman, L., Rousseeuw, P.J.: Finding groups in data: an introduction to cluster analysis, vol. 344. John Wiley & Sons (2009)
10. Kunze, M., Weidlich, M., Weske, M.: Behavioral similarity–a proper metric. In: BPM. pp. 166–181. Springer (2011)
11. Li, C., Reichert, M., Wombacher, A.: On measuring process model similarity based on high-level change operations. In: International Conference on Conceptual Modeling. pp. 248–264. Springer (2008)
12. Merelli, E., Rucco, M., Tesei, L., Piangerelli, M., Mamuye, A.L., Quadrini, M.: Survey of TOPDRIM applications of topological data analysis. In: Armano, G., Bozzon, A., Cristani, M., Giuliani, A. (eds.) KDWeb Workshop. CEUR Workshop Proceedings, vol. 1748. CEUR-WS.org (2016)
13. Ordoñez, A., Ordoñez, H., Corrales, J.C., Cobos, C., Wives, L.K., Thom, L.H.: Grouping of business processes models based on an incremental clustering algorithm using fuzzy similarity and multimodal search. Expert Systems with Applications **67**, 163–177 (2017)
14. Piangerelli, M., Maestri, S., Merelli, E.: Visualising 2-simplex formation in metabolic reactions. Journal of Molecular Graphics and Modelling **97** (2020)
15. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. of computational and applied mathematics **20**, 53–65 (1987)
16. Schoknecht, A., Thaler, T., Fettke, P., Oberweis, A., Laue, R.: Similarity of business process models—a state-of-the-art analysis. ACM Computing Surveys (CSUR) **50**(4), 1–33 (2017)
17. Schubert, E., Rousseeuw, P.J.: Faster k-medoids clustering: improving the pam, clara, and clarans algorithms. In: International conference on similarity search and applications. pp. 171–187. Springer (2019)
18. Van Der Aalst, W.: Data science in action. In: Process mining, pp. 3–23. Springer (2016)
19. Zhou, C., Liu, C., Zeng, Q., Lin, Z., Duan, H.: A comprehensive process similarity measure based on models and logs. IEEE Access **7**, 69257–69273 (2019)