novática
Revista de la Asociación de Técnicos de Informática

Novática, founded in 1975, is the oldest periodical publication amongst those especialized in Information and Communication Technology (ICT) existing today in Spain. It is published by **ATI (*Asociación de Técnicos de Informática*)** which also publishes **REICIS** (*Revista Española de Inovación, Calidad e Ingeniería del Software*).

<http://www.ati.es/novatica/>
<http://www.ati.es/reicis/>

ATI is a founding member of **CEPIS** (*Council of European Professional Informatics Societies*), the Spain´s representative in **IFIP** (*International Federation for Information Processing*), and a member of **CLEI** (Centro Latinoamericano de Estudios en Informática) and **CECUA** (*Confederation of EuropeanComputer User Associations*). It has a collaboration agreement with **ACM** (*Association for Computing Machinery*) as well as with diverse Spanish organisations in the ICT field.

Editorial Office, Advertising and Madrid Office
Plaza de España 6, 2ª planta, 28008 Madrid
Tlfn.914029391; fax.913093685 <novatica@ati.es>
Layout and Comunidad Valenciana Office
Av. del Reino de Valencia 23, 46005 Valencia
Tlfn. 963740173 <novatica_prod@ati.es>
Acounting, Subscriptions and Catalonia Office
Calle Avila 48-50, 3a planta, local 9, 08005 Barcelona
Tlfn.934125235; fax 934127713 <secregen@ati.es>
Andalucía Office <secreand@ati.es>
Galicia Office <secregal@ati.es>
Subscriptions and Sales <novatica.subscripciones@atinet.es>
Advertising Plaza de España 6, 2ª planta, 28008 Madrid
Tlnf.914029391; fax.913093685 <novatica@ati.es>
Legal depósit: B 15.154-1975 -- ISSN: 0211-2124; CODEN NOVAEC
Cover Page: "Mineral, Vegetable, Animal" - Concha Arias Pérez / © ATI
Layout Dising: Fernando Agresta / © ATI 2003

Special English Edition 2013-2014
Annual Selection of Articles

summary

Detection of Temporal Changes in Business Processes Using Clustering Techniques

Daniela Luengo, Marcos Sepúlveda
Computer Science Department, School of Engineering, Pontificia Universidad Católica de Chile, Santiago (Chile)

<dlluengo@uc.cl>,<marcos@ing.puc.cl>

Abstract: *Nowadays, organizations need to be constantly evolving in order to adjust to the needs of their business environment. These changes are reflected in their business processes, for example: due to seasonal changes, a supermarket's demand will vary greatly during different months of the year, which means product supply and/or re-stocking needs will be different during different times of the year. One way to analyze a process in depth and understand how it is really executed in practice over time, is on the basis of an analysis of past event logs stored in information systems, known as process mining. However, currently most of the techniques that exist to analyze and improve processes assume that process logs are in a steady state, in other words, that the processes do not change over time, which in practice is quite unrealistic given the dynamic nature of organizations. This document presents in detail the proposed technique and a set of experiments that reflect how our proposal delivers better results than existing clustering techniques.*

Keywords: *Concept Drift, Clustering, Process Mining, Temporal Dimension.*

Authors

Daniela Luengo was born in Santiago, Chile. She is an Industrial Civil Engineer with a major in Information Technology from Pontificia Universidad Católica de Chile. She also received the academic degree of Master of Science in Engineering from the same university. Additionally she has an academic certificate in Physical Activity, Sport, Health and Education. From 2009 to 2013 she worked at the Information Technology Research Center of the Pontificia Universidad Católica de Chile (CETIUC), as an analyst and consultant in the area of process management excellence, performing process mapping, process improvement and several researches. Her current interests are focused on applying her knowledge in the public sector of her country.

Marcos Sepúlveda was born in Santiago, Chile. He received his Ph.D. on Computer Science from the Pontificia Universidad Católica de Chile in 1995. He made a postdoctoral research in the ETH Zürich, Switzerland. He is an associate professor in the Computer Science Department at the Pontificia Universidad Católica de Chile since 2001. He is also the director of the Information Technology Research Center of his university (CETIUC). His research interests are Process Mining, Business Process Modeling, Business Intelligence, and Information Systems Management.

1. Introduction

In a globalized and hyper-connected world, organizations need to be constantly changing in order to adapt to the needs of their business environment, which implies that their business processes must also be constantly changing.

To illustrate this, consider the case of a toy store whose sales practices may change radically depending on whether they are executed over the Christmas period or during vacations, principally due to the changes in the volume of the demand. For the Christmas season, they might employ a process that prioritizes efficiency and volume (*throughput*), while during the vacation season they might use a process which focuses on the quality of their customer service.

In this example, it is easy to identify the periods in which demand changes, and thus it is possible that the process manager (the person in charge of the process management) will have a clear understanding of the changes that the sales process undergoes over time. However, if an organization has a process that occurs in various independent offices, for example, offices that are located in different geographic locations, the evolution of the changes in the process at each office will not be so evident to the central process manager. Moreover, the changes in each office could be different and could happen at different moments in time.

Understanding the changes that are occurring in the different offices could help to better understand how to improve the overall design of the process. Being able to understand these changes and model the different versions of the process allow the process manager to have more accurate and complete information in order to make coherent decisions which result in better service or efficiency.

To achieve the aforementioned goals, various advances have been made in the discipline of Business Process Management (BPM), a discipline which combines knowledge about information technology and management techniques, which are then applied to business operation processes, with the goal of improving efficiency [1]. Within BPM, process mining has positioned itself as an emerging discipline, providing a set of tools that help to analyze and improve business processes [1], based on analyzing event logs stored by information systems during the execution of a process. However, despite the advances made in this field, there still exists a great challenge, which consists of incorporating the fact that processes change over time, a concept which is known in the literature as *Concept Drift* [2].

Depending on the nature of the change, it is possible to distinguish different types of *Concept Drift*, including: *Sudden Drift* (sudden and significant change to the definition of the process), *Gradual Drift* (gradual change to the definition of the process, allowing for the simultaneous existence of the two definitions), and *Incremental Drift* (the evolution of the process that occurs through small, consecutive changes to the definition of the model).

Despite the existence of all these *Drift* types, the existing techniques of process mining are limited to finding the points in time when the process changes, centering principally on *Sudden Drift* changes. The problem with this limitation is that in practice it is not as frequent for business processes to show a sudden change in definition.

If we apply the existing process mining approaches to processes that have different kinds of changes other than *Sudden Drift*, we could end up with results which make little sense to the business.

In this document we propose a new approach, which allows the discovery of the various versions of a process when there are different kinds of *Drift*, helping to understand how the process behaves over time. To accomplish this task, existing process mining clustering techniques are used, but with time incorporated as an additional factor to the

> "Currently, one of the problems in process mining is that the developed algorithms assume the existence of information relative to a unique version of the process in the event log"

control-flow perspective to generate the different *clusters*. *Trace Clustering* techniques are used, which unlike other metrics-based techniques that measure the distance between complete sequences having linear complexity, allowing for the delivery of results in a shorter time span [3].

The focus of our work contributes to the process analysis, allowing the process manager to have a more realistic vision of how the process behaves over different periods of time. With this approach it is possible to determine the different versions of the process, the characteristics of each process, and to identify in which moment the changes occur.

This article is organized in the following way. Section 2 presents the related work. Section 3 describes base and the modified version of the clustering method. Section 4 presents experiments and results and finally the conclusions and future work are presented in section 5.

2. Related Work

Process mining is a discipline that has attracted major interest recently. This discipline assumes that the historical information about a process stored in information systems can be found in a dataset, known as an event log [4]. This event log contains past information about the activities that were performed in each step of the process, where each row of the log is composed of at least one identifier (id) associated with each individual execution of the process, the name of the activity performed, the timestamp (day and time when the activity occurred), and, optionally, additional information like the person who carried out the activity or other such information. Additionally, in the literature [3] an ordered list of activities invoked by a specific execution of the process is defined as a trace of the execution.

Currently, one of the problems in process mining is that the developed algorithms assume the existence of information relative to a unique version of the process in the event log. However, this is often not the case, which is why applying the process mining algorithms to these logs leads to fairly unrepresentative and/or very complicated results, which contribute little to the job of analyzing and improving the processes.

2.1. Clustering of the Event Log

To resolve the aforementioned problems in process mining, clustering techniques have been proposed for dividing the event log before the process mining techniques are applied [5]. This would consist of dividing the event log into homogenous clusters; in order to later apply the process mining techniques independently to each cluster and thus obtaining more representative models. Figure 1 shows the stage of log preprocessing and then uses a discovery technique as an example of a process mining technique.

To produce this clustering, it is necessary to define a way of representing the traces, so that it becomes possible to group them later according to previously determined criteria of similarity. There currently exist various clustering techniques in process mining [6][3]. The majority of them principally consider information about the control-flow of activities. These techniques can be classified into two categories:

1) Techniques that transform the traces into a vector space, in which each trace is converted into a vector. The dividing of the log can be done using a variety of clustering techniques in the vector space, like for example: *Bag of activities*, *K-gram model* [6], and *Trace clustering* [5]. However, these techniques have the problem of lacking contextual information, which some have attempted to correct with the *Trace clustering based on conserved patterns* technique [3].

2) Techniques that operate with the whole trace. These techniques use metrics of distance like *Levenshtein* and *Generic Edit Distance* [6], together with standard clustering techniques, assigning a cost to the difference between traces.

The existing techniques for both categories, despite improving clustering through the creation of structurally similar trace clusters,

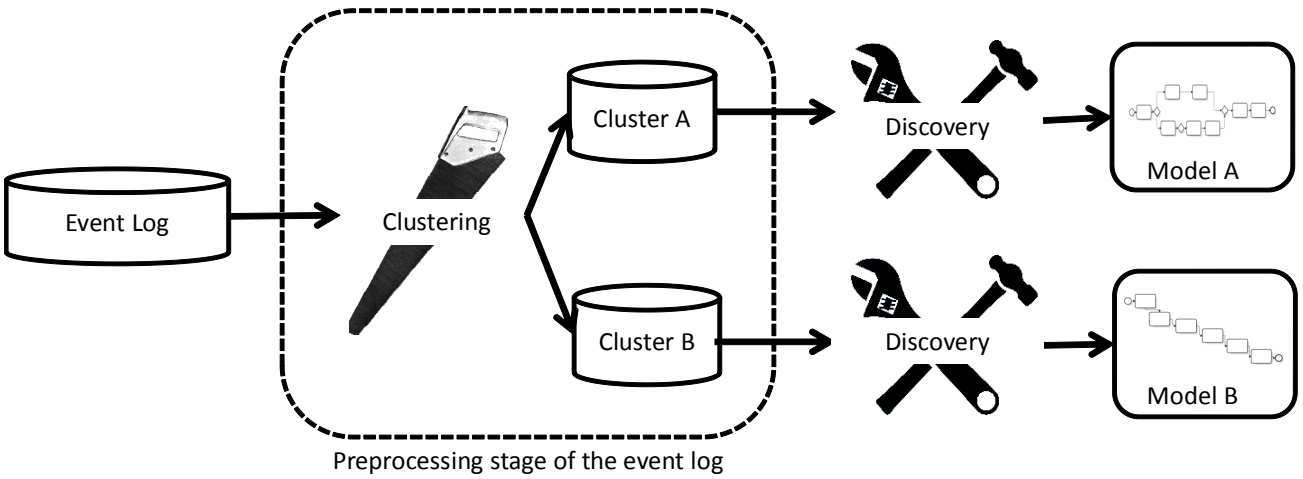

Figure 1. Preprocessing Stage of the Event Log.

> "The study of *Concept Drift* in the area of process mining has centered on process changes in the control-flow perspective, and can be of two kinds, momentary changes or permanent changes, depending on the duration of the change"

do not consider the temporal dimension of the process's execution, nor how the process changes over time.

2.2. The Concept Drift Challenge

In BPM, *Concept Drift* refers to a situation in which a process has experienced changes in its design within an analyzed period (yet the exact moment in which the changes were produced is unknown). These changes can be due to a variety of factors, but are mainly due to the dynamic nature of the processes [7].

The study of *Concept Drift* in the area of process mining has centered on process changes in the control-flow perspective, and can be of two kinds, momentary changes or permanent changes, depending on the duration of the change.

When changes occur over short and infrequent periods, they are considered momentary changes. These changes are also known in processes jargon as process noise or process anomalies.

On the other hand, permanent changes occur over more prolonged periods of time and/or when a considerable amount of instances is affected by the changes, which signals changes in the design process.

Our interest centers on the permanent changes in the control-flow perspective, which can be divided into the following four categories:

- Sudden Drift: This refers to drastic changes, meaning the way in which the process execution changes suddenly from one moment to the next.
- *Recurring Drift*: When a process suffers periodic changes, meaning a way of performing the process is repeated again later.
- *Gradual Drift*: This refers to changes which are not drastic, but rather at a moment when two versions of the process overlap, which corresponds to the transition from one version of the process to another.
- *Incremental Drift:* This is when a process has small incremental changes. These types of changes are more frequent in organizations that adopt agile BPM methodologies.

To solve the *Concept Drift* problem, new approaches have evolved to analyze the dynamic nature of the processes.

Bose [2] proposes methods to manage *Concept Drift* by showing how the process changes are indirectly reflected in the event log and that the detection of these changes is feasible by examining the relationship between activities. Different metrics have been defined to measure the relationship between activities. Based on these metrics, a statistical method was proposed whose basic idea is to consider a successive series of values and investigate if a significant difference between two series exists. If it does, this would correspond to a process change.

Stocker [8] also proposes a method to manage *Concept Drift* which considers the distances between pairs of activities of different traces as a structural feature, in order to generate chronologically subsequent *clusters.*

Bose and Stocker's approaches are limited to determining the moment in time when the process changes, and thus center on sudden changes and leave out other types of changes.

To resolve this, in an earlier article [9] we proposed an approach that makes use of clustering techniques to discover the changes that a process can experience over time, but without limiting ourselves to one particular kind of change. In that approach, the similarity among two traces is defined by the control-flow information and by the moment in which each trace begins to operate.

In this article, we present an extension of the earlier work [9], after incorporating a new form of measuring the distance between two traces.

Sequence	Maximal Repeat	Feature Set
bbbcd-bbbc-caa	{a, b, c, bb, bbbc}	{bb, bbbc}

Table 1. Example of *Maximal Repeat* and *Feature Set*.

3. Extending Clustering Techniques to Incorporate the Temporal Variable

As was explained in the last section, the existing approaches for dealing with *Concept Drift* are not sufficiently effective at finding the versions of a process when the process has undergone different types of changes. To solve this problem, we look to the *Trace Clustering* technique proposed by Bose [3] and based on conserved patterns, which allows clustering the event log considering each trace's sequence of activities.

Our work is based on this technique and extends it by incorporating an additional temporal variable to the other control-flow variables used for clustering.

3.1. *Trace Clustering* Based on Conserved Patterns

The basic idea proposed by Bose [3] is to consider subsequences of activities that repeat in multiple traces as feature sets for the implementation of clustering. Unlike the *K-gram* approach that considers subsequences of fixed size, in this approach the subsequences can be of different lengths. When two instances have a significant number of subsequences in common, it is assumed that they have structural similarity and these instances are assigned to the same cluster.

There are six types of subsequences, therefore, we will only give a formal definition of MR, since these are the subsequences that we used to develop our approach, however the work could be extended to use the other subsequences.

- Maximal Repeat (MR): A Maximal Repeat in a sequence T is defined as a subsequence that occurs in a Maximal Pair in T. Intuitively, an MR corresponds to a subsequence of activities that is repeated more than once in the log.

Table 1 shows an example where existing MR in a sequence are determined. This technique constructs a unique sequence starting from the event log, which is obtained by connecting all the traces, but with a delimiter placed among them. Then, the MR definition is applied to this unique sequence. The set of all MR discovered in the sequence with more than one activity, is called a *Feature set.*

> "The basic idea proposed by Bose is to consider subsequences of activities that repeat in multiple traces as feature sets for the implementation of clustering"

Based on the *Feature Set,* a matrix is created that allows the calculation of the distance between the different traces. Each row of the matrix corresponds to a trace and each column corresponds to a feature of the *Feature Set.* The values of the matrix correspond to the number of times that each feature is found in the various traces (see **Table 2**). We will call this matrix the Structural Features Matrix.

This pattern-based clustering approach uses "*Agglomerative Hierarchical Clustering*" as a clustering technique, with the minimum variance criterion [15], and using the Euclidian distance to measure the difference between two traces, defined in as follows:

$$dist(A,B) = \sqrt{\sum_{i=1}^{n}(T_{Ai} - T_{Bi})^2}$$

Where

dist (A, B) = distance between trace A and trace B.
n = number of features in de Feature Set.
T_{Ai} = number of times de feature in the appears in the trace A.

3.2. Clustering Technique to Include the Temporal Variable

To identify the various types of changes that can occur in business processes we must find a way to identify all the versions of a process. If we only look at the structural features (control-flow) then we leave out information regarding temporality. Both temporal and structural features are very important since the structure indicates how similar one instance is to another and the temporality shows how close in time the two instances are. Our approach looks to identify the different forms of implementing the process using both features (structural and temporal) at the same time, as is illustrated in **Figure 2**.

In order to mitigate the effects of external factors that are difficult to control, we use only the beginning of each process instance as the temporal variable.

For each trace, we store the time that have elapsed since a reference timestamp in the time dimension, for example, the number of days (or hours, minutes or seconds, depending on the process) elapsed since January 1st, 1970, to the timestamp in which the trace's first activity begins (see **Table 3**).

In this new approach, "*Agglomerative Hierarchical Clustering*" with the minimum variance criterion [15] is also used as a clustering technique.

To calculate the distance between two traces we use the Euclidian distance, but modified in order to consider at the same time the structural and temporal features.

First, we define T_{Ji} as the feature *i* of the trace *J*. If the feature *i* cannot be found in the trace *J*, its value will be 0, otherwise its value will be the number of times that the feature

Trace \ Feature set	bb	bbbc
bbbcd	2	1
bbbc	2	1
caa	0	0

Table 2. Structural Features Matrix.

i appears in the trace *J*.

$T_{J(n+1)}$ corresponds to the temporal feature of the trace *J* and its value is the number of days (or hours, minutes or seconds, depending on the process) that have elapsed since a reference timestamp. The index *(n+1)* is given to indicate that it is to be added to the structural features.

We define *L* as the set of all the *log* traces, and the expression $Max_{J\in L}(T_{Ji})$ represents the largest number of times the feature *i* appears in an event *log* trace. In the same way, $Min_{J\in L}(T_{Ji})$ corresponds to the smallest number of times the feature *i* appears in an event *log* trace.

$Min_{J\in L}(T_{J(n+1)})$ and $Max_{J\in L}(T_{J(n+1)})$ correspond to the earliest and latest time in which an event log trace begin. Also we define $D_E(A,B)$ and $D_T(A,B)$ as the structural and temporal distance between the trace A and the trace B, respectively.

$$D_E(A,B) = \sqrt{\sum_{i=1}^{n}\left(\frac{T_{Ai} - T_{Bi}}{\max_{J\in L}(T_{Ji}) - \min_{J\in L}(T_{Ji})}\right)^2}$$

$$D_T(A,B) = \sqrt{\left(\frac{T_{A(n+1)} - T_{B(n+1)}}{\max_{J\in L}(T_{J(n+1)}) - \min_{J\in L}(T_{J(n+1)})}\right)^2}$$

where:

n = number of features from the Structural Features Set.

Even though both distances, D_E and D_T are normalized, since the domain of D_E is greater than the one of D_T, we define Min_E, Max_E, Min_T and Max_T as:

$$Min_E = \min_{A,B\in L}\sqrt{D_E(A,B)} \quad , \quad A \neq B$$

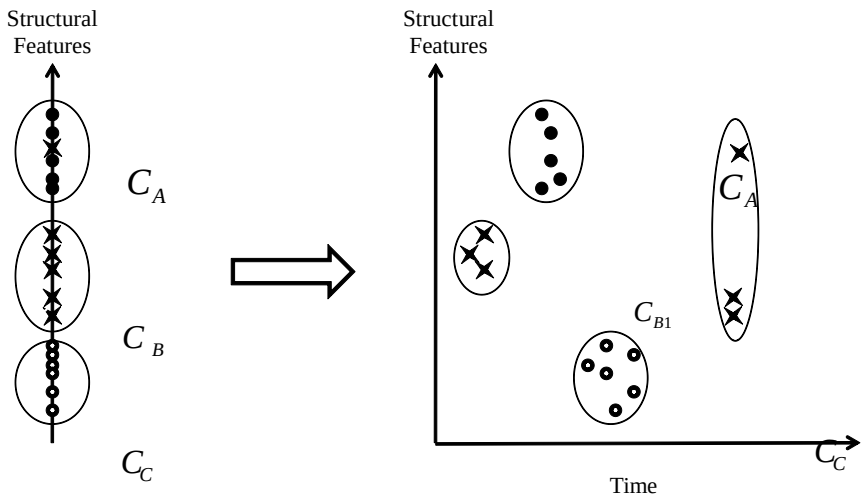

Figure 2. Example of the Relevance of Considering Time in the Analysis.

$$Max_E = \max_{A,B \in L} \sqrt{D_E(A,B)} \quad , \quad A \neq B$$

$$Min_T = \min_{A,B \in L} \sqrt{D_T(A,B)} \quad , \quad A \neq B$$

$$Max_T = \max_{A,B \in L} \sqrt{D_T(A,B)} \quad , \quad A \neq B$$

Min_E and Max_E correspond to the minimum and maximum (normalized) distance between all traces, considering only structural features.

Min_T and Max_T correspond to the minimum and maximum (normalized) distance between all traces, considering only temporal features.

The new way of measuring the distance between two traces, $dist(A,B)$, incorporates the parameter μ, which we will call the weight of the temporal dimension, which serves to weigh the structural and temporal features. Additionally, this new way of measuring the distance adjusts D_E and D_T in such a way that the weight of both distances are equivalent.

$$dist(A,B) = (1-\mu)\frac{D_E(A,B) - Min_E}{Max_E - Min_E} + \mu\frac{D_T(A,B) - Min_T}{Max_T - Min_T}$$

The weight of the temporal dimension, μ, can have values between 0 and 1, according to the relevance given to the temporal feature.

4. Evaluation

We analyzed the proposed technique using six event logs obtained from different synthetic processes, which were created with CPN Tools [10][11]. In order to measure their performance we used the *Guide Tree Miner* plug-in [3] available in ProM 6.1 as well as a modified version of this plug-in that incorporates the proposed changes.

The evaluation was carried out using different metrics to measure the new approach's classification effectiveness versus the base approach.

4.1. Experiments and Results

Figure 3 shows the sequence of steps performed in the experiments.

1) To create the synthetic *log*, a simulation was used based on two designed models, M1 and M2, using CPN Tools.
2) The method starts by applying the clustering technique on the synthetic *log* received, which generates a given number of *clusters*. In this case, two *clusters* (C_A *and* C_B). Two clustering techniques are applied:

- **Base approach:** *Trace clustering* based on conserved patterns.
- **Our approach:** *Extended trace clustering*, where the temporal dimension is incorporated. In this technique, the weight of the temporal dimension, μ, can be given different values, which vary between 0 and 1.

3) For each of the clusters a discovery process is carried out, generating two new models (M_A *and* M_B).

The approach's performance can be measured at two points:
a) Conformance 1: The metrics are measured between any of the original models and one of the generated *clusters*.
b) Conformance 2: Metrics are measured between any of the original models and one of the generated models.

The metrics used to measure Conformance 1 are the following:

- *Accuracy*: Indicates the number of instances correctly classified in each *cluster*, according to what is known about the original events *log*. These values are between 0% and 100%, where 100% indicates that the clustering was exact.
- *Fitness*: Indicates how much of the observed behavior in an event *log*, (for example, *cluster* C_A) is captured by the original process model (for example, model M_2) [12]. These values are between 0 and 1, where 1 means the model is capable of representing all the *log* traces.
- Precision: Quantifies whether the original model allows for behavior completely unrelated to what is seen in the event *log*. These values are between 0 and 1, where 1 means that the model does not allow behavior additional to what the traces indicate [13].

The metrics used to measure Conformance 2 are the following [14]:

- *Behavioral Precision* (B_P)
- *Structural Precision* (S_P)
- *Behavioral Recall* (B_R)
- *Structural Recall* (S_R)

These metrics quantify the precision and generality of one model with respect to another. The values of these four metrics are between 0 and 1, where 1 is the best possible value.

Table 4 summarizes the results of applying the base approach and the new approach (varying the value of the parameter μ), to the different synthetic event *logs* created. This table shows the *accuracy* metric, which indicates the percentage of correctly classified traces.

For each *log*, the highest *accuracy* is reached with our approach, but with different μ values (varying between 0.2 and 0.9). The *accuracy* varies with different μ values because in each *log* the relevance of the temporal dimension versus the structure of the process is not the same.

We use Log (f) to make a more in-depth analysis of the results, measuring all the metrics defined both in Conformance 1 and Conformance 2 (see **Table 5**).

All metrics calculated for Log (f) show good results when the parameter μ has a value of 0.5 or 0.6. When the five metrics are averaged, the highest overall average is obtained with μ equal to 0.5.

5. Conclusions and Future Work

In this article we present the limitations of current clustering techniques for process mining, which center on grouping structurally similar executions of a process. By focusing just on the structure of the executions, the process's evolution over time (*Concept Drift*) is left out. New techniques have been developed to address this, but these also present limitations since they focus on finding the points in which the process changes, which is limited to just one kind of change, *Sudden Drift*.

We present an approach that extends current clustering techniques in order to find the different versions of a process that changes overt time (in multiple ways, i.e., different types of *Concept Drifts*), allowing for a better understanding of the variations that occur in the process and how, in practice, it is truly being performed over time.

Our work focuses on the identification of models associated with each version of the process. The technique we propose is a tool that helps business managers to make decisions. For example, it can help determine if the changes produced in the implementation of the process are really those that are expected, and based on this, take the proper action if they discover abnormal behaviors. Also, by understanding and comparing the different versions of a process, good and bad practices can be identified, which are highly useful when the time comes to improve or standardize the process.

In this document we present a set of metrics to measure the performance of our approach, i.e., whether the approach is able to cluster data in the same way the data was created. Thus, these metrics require *a priori* process information, which is not feasible for real cases.

Each metric measures a different aspect, which when used together, allow for a multi-

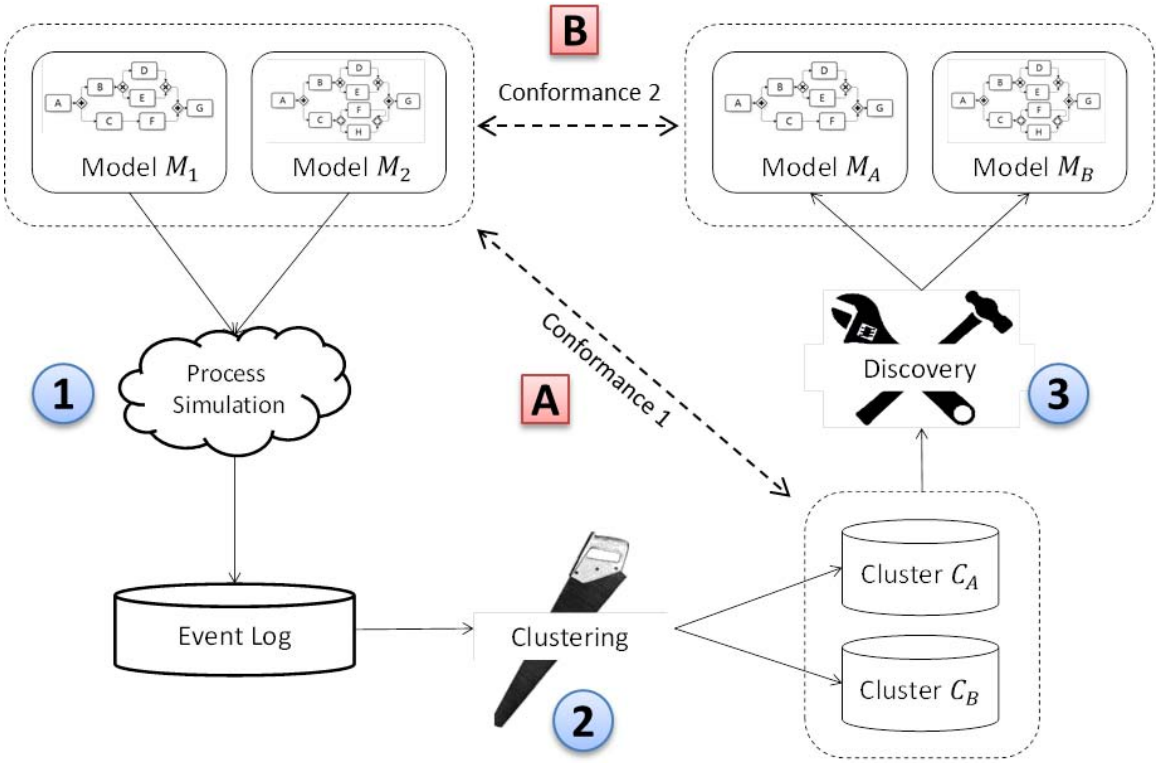

Figure 3. Steps for Performing the Experimental Tests.

Approach	μ	Log (a)	Log (b)	Log (c)	Log (d)	Log (e)	Log (f)
Base	-	57%	38%	55%	86%	99%	53%
New	0.0	59%	52%	49%	82%	59%	51%
	0.1	65%	52%	49%	82%	59%	68%
	0.2	87%	52%	63%	100%	59%	64%
	0.3	87%	52%	63%	100%	59%	62%
	0.4	95%	52%	63%	100%	59%	63%
	0.5	100%	52%	63%	100%	59%	95%
	0.6	100%	52%	73%	100%	100%	94%
	0.7	96%	89%	73%	100%	100%	67%
	0.8	78%	88%	73%	74%	84%	55%
	0.9	79%	77%	77%	68%	77%	78%
	1.0	81%	78%	68%	73%	72%	55%

Table 4. *Accuracy* Metric Calculated for the Six Synthetic Test *Logs*.

Approach	μ	Accuracy	Fitness	Precision	Average B_P and S_P	Average B_R and S_R	Overall Average
Base	-	53%	0.93	0.78	0.77	0.81	0.76
	0.0	51%	0.93	0.73	0.73	0.70	0.72
	0.1	68%	0.93	0.81	0.86	0.86	0.83
	0.2	64%	0.92	0.80	0.83	0.82	0.80
	0.3	62%	0.92	0.80	0.80	0.78	0.78
	0.4	63%	0.92	0.79	0.83	0.86	0.81
New	0.5	95%	0. 95	0.85	0.97	0.96	0. 94
	0.6	94%	0.95	0.86	0.95	0.94	0.93
	0.7	67%	0.92	0.84	0.84	0.89	0.83
	0.8	55%	0.94	0.87	0.88	0.94	0.84
	0.9	78%	0.94	0.81	0.89	0.95	0.87
	1,0	55%	0.93	0.87	0.88	0.95	0.84

Table 5. Different Metrics for Analyzing Log (f)

faceted vision that makes the analysis more complete.

One key aspect of our clustering approach is the value it is given to the weight of the temporal dimension, parameter μ, which is closely related to the nature of the process. High μ values give a greater importance to time when carrying out the clustering, whereas low μ values give more importance to the structural features of the process.

The experiments results show that the approach proposed in this document has a better performance and that exists at least a value for the parameter μ that gives better results in comparison to only using the structural clustering technique (*Trace clustering* based on patterns). This is achieved because our approach is capable of grouping the *log* traces in such a way so as to identify structural similarity and temporal proximity at the same time.

One of the metrics used is *accuracy*. In some experiments, this metric reached 100%, meaning all traces were classified correctly. When a 100% *accuracy* was not reached, it was because there are processes whose different versions are similar amongst themselves, and therefore there are traces that can correspond to more than one version of the process, which makes the classification not exactly the same as what is expected.

Our future work in this line of investigation is to test the new approach with real processes. We also want to work on developing the existing algorithms so that they are capable of automatically determining the optimal number of *clusters*. In order to do so, it will be necessary to define new metrics that will allow us to calculate the optimal number of clusters without *a priori* information of the process versions.

References

[1] W. van der Aalst. *Process Mining, Discovery, Conformance and Enhancement of Business Processes.* Springer, 2011. ISBN 978-3-642-19345-3.

[2] R.J. Bose, W. van der Aalst, I. Zliobaite, M. Pechenizkiy. Handling Concept Drift in Process Mining. *23rd International Conference on Advanced Information Systems Engineering.* London, 2011.

[3] R. Bose, W. van der Aalst. Trace Clustering Based on Conserved Patterns : Towards Achieving Better Process Models. *Business Process Management Workshops*, pp. 170-181, 2010. Berlin: Springer Heidelberg.

[4] W. van der Aalst, B. van Dongen, J. Herbst, L. Maruster, G. Schimm, A. Weijters. *Data & Knowledge Engineering*, pp. 237-267, 2003.

[5] M. Song, C. Günther, W. van der Aalst. Trace Clustering in Process Mining. *4th Workshop on Business Process Intelligence (BPI 08)*, pp. 109-120. Milano, 2009.

[6] R. Bose, W. van der Aalst. Context Aware Trace Clustering: Towards Improving Process Mining Results. *SIAM*, pp. 401-412, 2009.

[7] W. van der Aalst, A. Adriansyah, A.K. Alves de Medeiros, F. Arcieri, T. Baier, T. Blickle et al. *Process Mining Manifesto*, 2011.

[8] T. Stocker. Time-based Trace Clustering for Evolution-aware Security Audits. *Proceedings of the BPM Workshop on Workflow Security Audit and Certification*, pp. 471-476. Clermont-Ferrand, 2011.

[9] D. Luengo, M. Sepúlveda. Applying Clustering in Process Mining to find different versions of a business process that changes over time. *Lecture Notes in Business Information Processing*, pp. 153-158, 2011.

[10] A.V. Ratzer, L. Wells, H.M. Lassen, M. Laursen, J. Frank, M.S. Stissing et al. CPN Tools for editing, simulating, and analysing coloured Petri nets. *Proceedings of the 24th international conference on Applications and theory of Petri nets* pp. 450-462. Eindhoven: Springer-Verlag, 2003.

[11] A.K. Alves De Medeiros, C. Günther. Process Mining: Using CPN Tools to Create Test Logs for Mining Algorithms. *Proceedings of the Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pp. 177–190, 2005.

[12] A. Rozinat, W. van der Aalst. Conformance testing: Measuring the fit and appropriateness of event logs and process models. *Business Process Management Workshops*, pp. 163-176, 2006.

[13] J. Muñoz-Gama, J. Carmona. A fresh look at precision in process conformance. *Proceeding BPM'10, Proceedings of the 8th international conference on Business process management*, pp. 211-226, 2010.

[14] A. Rozinat, A.K. Alves De Medeiros, C. Günther, A. Weijters, W. van der Aalst. Towards an Evaluation Framework for Process Mining Algorithms. *Genetics,* 2007.

[15] J. Ward. Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, pp. 236-244, 1963.