Temporal analysis of process logs: a case study

Michele Berlingerio[1,2] Fosca Giannotti[2] Mirco Nanni[2] Fabio Pinelli[2]

[1]IMT School for Advanced Studies piazza San Ponziano, 6 Lucca, Italy
[2]KDD-Lab ISTI-CNR via G. Moruzzi, 1 Pisa, Italy
email: {name.surname}@isti.cnr.it

Extended Abstract

Abstract. In the last years there has been increasing interest to the analysis of process logs and several techniques such as workflow mining have been proposed aimed at automatically deriving the underlying workflow models. However, current approaches essentially disregard an important information contained in these traces, the timestamps, used only to define a sequential ordering of the performed tasks.
In this work we try to overcome these limitations, by explicitly including the temporal information in the analysis, that allows to distinguish among different temporal behaviours. That makes it possible to discern between apparently identical process executions that are performed with different transition times between consecutive tasks. Detecting such differences would enable a more accurate comparison between the original workflow design and its actual usage.
This work faces the above problem by applying a novel mining paradigm named Time-Annotated Sequences ($\mathcal{TAS}$) aimed at extracting sequential patterns where each transition between two events is annotated with a typical transition time that emerges from input data. We report a real-world case study, in which the $\mathcal{TAS}$ mining paradigm is applied to databases of log traces obtained from the actual execution of processes. We believe that this case study not only shows the interestingness of extracting $\mathcal{TAS}$ patterns in the workflow context, but, more ambitiously, it opens the way for a novel approach to workflow mining.

1 Introduction

In the last years, most organizations started to use information systems to support the execution of their business processes[13]. With the increasing number of these available systems, the volume of the available collected processes logs is increasing very fast. Such logs constitute a very important information in several contexts: for example, in the context of logistics, the optimization of times is crucial, every step should be made strictly on time, and if there are anomalies or problems, the entire logistic solution should be redesigned.
For such reasons, the importance of analysing the process logs has been increasing very fast in the last years[15, 14, 7]. However, analyzing such logs can be hard from different point of views: too many data, too complicated original process

diagram, too many users to check.
Thus, in the last years, several techniques such as workflow mining have been proposed aimed at automatically deriving the workflow models starting from the process logs [16, 17, 11]. However, current approaches use the temporal information contained in the logs only for keeping track of the temporal order of the performed tasks. By the way, the timestamps contained in the logs reflects a much richer information that allows to distinguish among different temporal behaviours. For example, suppose we have tasks A, B and C to be executed, and suppose that the transition time from A to B is usually 1 minute, and 9 minutes from B to C; if we have an execution with 9 minutes from A to B and 1 minute from B to C then we are in presence of an anomaly during the process, even if the sequence of the performed tasks follows the process workflow. In this case, the usual workflow mining techniques do not detect the anomaly and treat the abnormal execution as a normal one. Neither the use of some Sequential Patterns Mining technique can help in this direction.
The main contribution of this paper is to provide a methodological approach to this kind of data, by means of Time-Annotated Sequences ($\mathcal{TAS}$) mining [8, 9]. $\mathcal{TAS}$ patterns extraction is a novel mining paradigm defined in our laboratory together with an efficient algorithm for extracting them and recently successfully applied to biological data [5, 4]. $\mathcal{TAS}$ are sequential patterns where each transition between two events is annotated with a typical transition time that is found frequent in the data. In principle, this form of pattern is useful in several contexts: for instance, (i) in web log analysis, different categories of users (experienced vs. novice, interested vs. uninterested, robots vs. humans) might react in similar ways to some pages - i.e., they follow similar sequences of web access - but with different reaction times; (ii) in medicine, reaction times to patients' symptoms, drug assumptions and reactions to treatments are a key information; (iii) in workflow logs, the typical data is a sequence of operations performed at specific moments and could be interesting to look for frequent sequences having frequent temporal annotations. In all these cases, enforcing fixed time constraints on the mined sequences is not a solution. It is desirable that typical transition times, when they exist, emerge from the input data. $\mathcal{TAS}$ patterns have been also used as basic brick to build a truly spatio-temporal trajectory pattern mining framework [10].
The rest of the paper is organized as follows. In Section 2 we summarize some work related to this paper. In Section 3 we introduce the $\mathcal{TAS}$ paradigm and how this can be used for mining time-annotated data. Section 4 describes the real-world case study in which the $\mathcal{TAS}$ paradigm is applied to a real-world and a syntectic dataset. Section 5 summarizes the contributions and the results of this paper, and draws some future lines of research.

2 Related Work

In [12], Hwang et al. propose a method for discovering temporal patterns from process instances. They divide the temporal relationships between any activity pair in *followed* and *overlapped*, by analysing the starting and ending times of each activity. After having traversed the activities in the given process instance

by the ascending order of their starting times, they create the associated *temporal graph*. Given the usual definitions of support, frequency and maximality, and a set of process instances, the *temporal pattern discovery* problem defined here is to find the maximal temporal graphs among all frequent temporal graphs. Three different algorithm, *TP-Graphs*, *TP-Itemset* and *TP-Sequence* are proposed for solving such a problem, working with graphs, items and sequences, respectively. These algorithms are all based on the Apriori[2] idea. However, in this work it is clear how the temporal information is only used for ordering the tasks in the process instances: in the temporal graph pattern itself there is explicit temporal information. So, there is no focus, for example, on frequent execution or transition times.

3 The TAS mining paradigm

In this section we briefly summarize the main aspects of the $\mathcal{TAS}$ mining paradigm.

The $\mathcal{TAS}$ is a form of sequential patterns annotated with temporal information representing typical transition times between the events in a frequent sequence that was introduced in [9]. The $\mathcal{TAS}$ mining paradigm finds all the sequences of the form

$$T = s_0 \xrightarrow{t_1} s_1 \xrightarrow{t_2} ... \xrightarrow{t_n} s_n$$

meaning that the sequence $s_0...s_n$ occurs in at least a fraction σ of the input sequences, with transition times *similar* to $t_1...t_n$, where two transition time are considered similar if their difference is not larger than a given threshold τ.

To better understand the results shown in section 4, consider an example pattern resulting from the execution of MiSTA 1.1[8], the implementation of the $\mathcal{TAS}$ paradigm: (Construction) (Building Up) (Released), support: 115128, temporal annotations: [0,60], [68,124]. This resulting pattern can be viewed as:

$$(Construction) \xrightarrow{[0,60]} (Building\ Up) \xrightarrow{[68,124]} (Released)$$

and can be read as: the sequence *Construction, Building Up, Released* is frequently executed with a support of *115128*, with typical transition times of $t_1 \in [0, 60]$ for the first transition and $t_2 \in [68, 124]$ for the second one.

For more details see [8, 9].

4 Case Study: mining TAS from process log data

In this section we describe the results obtained applying the $\mathcal{TAS}$ paradigm to process logs data.

4.1 Introduction to Process Logs

In this paper we applied the $\mathcal{TAS}$ mining paradigm to process logs. This kind of automatically collected data usually contains information about cases (i.e. process instances) that have been executed in an organization, the times at which the tasks were executed, the persons or systems that performed these tasks, and other kinds of data. These logs are the starting point for process mining, and

are usually called *event logs* or *workflow logs*. The type of information contained in the logs determines which type of process mining can be performed. If the log (i) provides the tasks that are executed in the process and (ii) it is possible to infer their order of execution and link these taks to the process instances, then the control-flow perspective can be mined.

With this kind of data there are several possible scenarios:

1. it is possible to group the data by instance ids and, with the help of $\mathcal{TAS}$ mining, to extract frequent patterns that give an idea of the transition times: this could be useful for finding sequences of tasks which are bottlenecks for the process
2. it is possible to group the data by users and to extract patterns that give an idea of the execution speed of each user: this could be useful for finding groups of users that execute sequences of tasks too fast / too slowly
3. in any case it is possible to perform anomaly and regularities detection: it could be useful to look for sequences of tasks that are executed too slowly / too fast
4. it is possible to determine the distribution of the transitions times between the tasks: it could be useful for understanding if (a part of) the process can be (or has to be) redesigned for achieving better performance

Note that under certain circumstances, $\mathcal{TAS}$ are the only possible method for looking for frequent sequences, since the amount of data can make the problem unfeasible for most of the sequential pattern miners, and the proper use of the τ parameter can make the difference in that way [8, 9].

In the following sections we present the datasets we worked on, and the results of applying the $\mathcal{TAS}$ mining framework on them. We present some results covering each of the above scenarios, to better show the meaning of the $\mathcal{TAS}$ mining paradigm.

4.2 Datasets

We worked on two process logs datasets: the first one comes from the usage of a real-world system developed by Think3[1], which is an object repository managing system, that allows the users to operate on the same objects from different locations; the second dataset is the one relative to the ProM tutorial[6], and contains syntectic logs from a technical assistance chain process of a phone company. From now on we refer at them as the Think3 and the ProM datasets, respectively. The timestamps contained in the logs represent the exact moment in which the event occurred. In particular, in the Think3 dataset, we did not have the starting and ending time of an operation, so we assumed that they are instantaneous and that the timestamps generally refer to the pair (execution time, transition time). In the ProM dataset, the operations are clearly divided in "started" and "ended", so we intended all the timestamps as transition times between events.

4.3 Tools

We performed the experiments using the implementations of two algorithms: MiSTA 1.1[8] for the extraction of $\mathcal{TAS}$ and SPAM 1.3.3[3] for the extraction of sequential patterns. We performed all the experiments on an Intel Mac Core Duo 2.16ghz, equipped with 1gb ram, running Mac Osx10.4.

4.4 Results and evaluation

We divided the entire analysis in several steps, to be able to show some interesting results for each of the key points listed in section 4.1: first we performed a simple comparison between the extraction of sequential patterns and the extraction of $\mathcal{TAS}$. The aim of this step was to show the difference in the resulting patterns, and how the $\mathcal{TAS}$ can be more informative and useful. Then we grouped the data of the two datasets by instance, so that every transaction is a process instance, viewed as a (long) sequence of performed tasks, and we looked for the frequent temporally-annotated sequences among them. This step was aimed at looking for regularities in the transition times between tasks, in such a way to be able to determine if there are anomalies or bottlenecks along the process. Subsequently, we grouped the data by users and executed MiSTA on them: in this way we were able to see the regularities on the execution speeds of the users, in such a way that it is possible to determine users that execute tasks too slowly, or if there are users that execute tasks in a very irregular way. Then we changed the τ parameter to several different values, to show how it is possible to find sequences with transition times having high or low variance by using $\mathcal{TAS}$.
The rest of this section shows the results obtained following the above steps.

ID	Pattern	Support (%)	Support (abs)	Temporal Annotations
Think3				
1	(Construction) (Building_Up) (Released)	37.8	115128	[0,60] [0,60] [0,60] [68,124] [0,60] [132,188]
SPAM found the same pattern with 119896 occurrences				
ProM				
2	(TestRepair_start) (TestRepair_complete) (ArchiveRepair_complete)	69.6	1464	[430,470] [490,530] [430,470] [550,590] [490,530] [490,530] [490,530] [550,590] [490,530] [310,350]
SPAM found the same pattern with 1996 occurrences				

Table 1. Some results of the first step

First step. Table 1 shows the results of the first step. In column 1 we have the ID of the pattern; in column 2 the frequent pattern found; in column 3 the support of the pattern expressed in percentage (please note that this is the support of the sequence part of the pattern, for which several frequent annotations where found); in column 4 the same support in absolute value; in column 5 temporal annotations of the mined $\mathcal{TAS}$, according to section 3. The temporal annotations are expressed in seconds elapsed between two consecutive tasks.
In this step, we simply performed a comparison between the information extracted by frequent sequential pattern mining and frequent temporally-annotated sequences, to show the different kind of interestingness of the resulting patterns. As one can see in Table 1, using the temporal dimension not only drastically reduces the support of the patterns (i.e.: returning more informative patterns), but also allows the user to explore several different frequent temporal annotations. This means more information, which is also more usable (less patterns,

more informative, easier to explore). Note that we set the same minimum support for MiSTA and SPAM, but the τ parameter played here an important role: the lesser is τ, the fewer are the frequent patterns found.

ID	Pattern	Support (%)	Support (abs)	Temporal Annotations
Think3				
1	(Released) (Administrator) (Released)	24	7327	[504,1800] [0,8] [504,1800] [8,120] [248,1800] [1672,1736]
ProM				
2	(Repair(Simple)_start) (Repair(Simple)_complete) (TestRepair_start)	39	822	[1900,1940] [0,200] [280,500] [1660,1700] [460,500] [1300,1340]
3	(Register_complete) (AnalyzeDefect_start) (TestRepair_start) (TestRepair_complete)	83	1747	[0,200] [2200,2240] [500,560] [0,200] [3400,3440] [400,500] [0,200] [1660,1700] [340,400]

Table 2. Some results of the second step

Pattern	τ	Sequence is frequent?	# Temporal annotations
Think3			
(Construction) (Building_Up) (Released)	<600	Yes	0
	600	Yes	5
	1800	Yes	52
	3600	Yes	107
ProM			
(TestRepair_start) (TestRepair_complete) (ArchiveRepair_complete)	<600	Yes	0
	600	Yes	18

Table 3. Some results of the third step

Second step. Table 2 shows some of the results of the second step of analysis (the format is the same of Table 1). In this sample results, it is clear how it is possible to distinguish between tasks that are executed with a low transition time and the ones that are executed with a high transition time. For example, consider pattern #1: the time between (Released) and (Administrator) is much higher than the time elapsed between (Administrator) and (Released), at least in 2 of the 3 temporal annotations reported in the table. Instead, we can see how the third temporal annotation denotes a different behaviour along the process. This could mean that sometimes the second transition is performed quickly, whilst sometimes it can also be a bottleneck for the process. To better understand this situation, one should have the designed (or expected) transition times among the process and see wether there is the presence of anomalies or not. Considerations like these can be also done for the patterns #2 and #3 found in the other dataset. Note also that while in pattern #1 and pattern #2 we found some frequent temporal annotation that seems to show temporal anomalies during the process, in pattern #3 the temporal annotations are rather similar to each other. Here we are in a clear presence of strong regularities in the transition times: maybe it could mean that if those times were not the expected (or allowed) ones, this is

clearly one situation where the entire process should be redesigned or optimized (Performance Analysis).

We also performed this analysis but firstly grouping the data by users, so that every transaction is the sequence of performed tasks per single user. Repeating this analysis in this kind of preprocessed data gave the same kind of results (and considerations) as above: it is possible to find out that some user regularly perform tasks slowly (i.e.: they are the "bottlenecks" of the process), or anomalies during the execution of the process by a group of users (meaning a possible problem encountered during the process by that group). Unfortunately, due to the statistics of the two datasets, it was quite difficult to put the very long resulting sequences on a table.

Third step. Table 3 shows the results of the third step of analysis. In column 1 we have a sample extracted pattern; in column 2, the τ threshold value that was used to generate the pattern; in column 3 we specify if the sequence was found frequent in the data (note that the value of this column is always "yes", but in both the first rows of each of the two patterns we did not find any frequent temporal annotation); in column 4 we have the number of temporal annotations found for the pattern.
This step is rather different from the previous ones: as one can immediately see, we tried several different values of the τ parameter to show the relationship between it and the frequent temporal annotations found. This step shows a particular usage of the τ parameter: we can see that the higher is the value of τ, the higher is the number of the temporal annotations found. This is a straightforward implication of the $\mathcal{TAS}$ definition, and it is trivial for the user to understand this key point. The interesting fact is that here we have a measure of the distribution of the transition times along the process. This is an easy way for quickly detect if the transition times present some regularities (fewer temporal annotations mean more regular transition times) among the process, aiming at checking the optimality of the followed workflow.

5 Conclusions and Future Work

We have described a pattern discovery analysis that we conducted on both a real-word and a synthetic process log datasets. The aim of the data analysis was assessing the adequateness of the $\mathcal{TAS}$ mining paradigm to this kind of data.
After giving the necessary introduction to the $\mathcal{TAS}$ mining paradigm, we have stated a list of possible open problems and issues related to process log mining. We have presented the results achieved to these problems, with particular focus on Performance Analysis issues, where the temporal dimension plays an important role as a measure of optimality. The contribution of this paper as case study is thus mainly methodological.
In our future work we are going to investigate the possibility of building a $\mathcal{TAS}$-based mining framework that solves an extended version of the workflow mining problem, in which the workflow diagram is enriched by temporal constraints on the transition arcs between consecutive tasks.

References

1. The think3 company. http://www.think3.com.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.
3. J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 429–435, New York, NY, USA, 2002. ACM Press.
4. M. Berlingerio, F. Bonchi, F. Giannotti, and F. Turini. Mining clinical data with a temporal dimension: a case study. In *Proceedings of The 1st IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2007)*, November 2007.
5. M. Berlingerio, F. Bonchi, F. Giannotti, and F. Turini. Time-annotated sequences for medical data mining. In *Proceedings of The IEEE International Workshop of Data Mining in Medicine 2007 (DMMed '07 in conjunction with ICDM'07)*, October 2007.
6. A. K. A. de Medeiros and A. T. Weijters. ProM Framework Tutorial. http://www.processmining.org, November 2006.
7. P. L. (editor). *Workflow Handbook 1997, Workflow Management Coalition.* John Wiley and Sons, New York, 1997.
8. F. Giannotti, M. Nanni, and D. Pedreschi. Efficient mining of temporally annotated sequences. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, 2006.
9. F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli. Mining sequences with temporal annotations. In *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC)*, pages 593–597, 2006.
10. F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli. Trajectory patter mining. In *The Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007.
11. G. Greco, A. Guzzo, L. Pontieri, and D. Saccà. Discovering expressive process models by clustering log traces. *IEEE Trans. Knowl. Data Eng.*, 18(8):1010–1027, 2006.
12. S.-Y. Hwang, C.-P. Wei, and W.-S. Yang. Discovery of temporal patterns from process instances. *Comput. Ind.*, 53(3):345–364, 2004.
13. H. Ma. Process-aware information systems: Bridging people and software through process technology: Book reviews. *J. Am. Soc. Inf. Sci. Technol.*, 58(3):455–456, 2007.
14. S.Jablonski and C.Bussler. *Workflow Management: Modeling Concepts, Architecture and Implementation*. International Thomson Computer Press, 1996.
15. W. M. P. van der Aalst, J. Desel, and A. Oberweis, editors. *Business Process Management, Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*. Springer, 2000.
16. W. M. P. van der Aalst, B. F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A. J. M. M. Weijters. Workflow mining: a survey of issues and approaches. *Data Knowl. Eng.*, 47(2):237–267, 2003.
17. T. Weijters and W. M. P. van der Aalst. Process mining: Discovering workflow models from event-based data. In Krse, B., Rijke, M., Schreiber, G., and Someren, M., editors, *Proceedings of the 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2001)*, pages 283–290, 2001.