Adapting Association Rule Mining to Discover Patterns of Collaboration in Process Logs

Stefan Schönig
University of Bayreuth
Bayreuth, Germany
stefan.schoenig@uni-bayreuth.de

Michael Zeising
University of Bayreuth
Bayreuth, Germany
michael.zeising@uni-bayreuth.de

Stefan Jablonski
University of Bayreuth
Bayreuth, Germany
stefan.jablonski@uni-bayreuth.de

Abstract— **The execution order of work steps within business processes is influenced by several factors, like the organizational position of performing agents, document flows or temporal dependencies. Process mining techniques are successfully used to discover execution orders from process execution logs automatically. However, the methods are mostly discovering the execution order of process steps without facing possible coherencies with other perspectives of business processes, i.e., other types of process execution data. In this paper, we propose a method to discover cross-perspective collaborative patterns in process logs and therefore strive for a genotypic analysis of recorded process data. For this purpose, we adapted the association rule mining algorithm to analyse execution logs. The resulting rules can be used for guiding users through collaborative process execution.**

Keywords—Process Mining, Data Mining, Association Rule Mining, Business Rules, Guidance through Process Execution

I. Introduction

In our daily lives we are involved in a variety of processes, for example when we book a trip via the internet [1]. More and more information about these processes is recorded in the form of process execution logs. Information systems record a great variety of process execution information and data. This process data can be used to reconstruct underlying process models or to identify previously unknown patterns within the recorded information.

Using *process mining* techniques, it is possible to discover process models automatically from process execution logs [2]. However, existing approaches have a common drawback: they are mainly examining the execution order of process steps and therefore focus on a *phenotypic* analysis, i.e., reproducing the things that happened, of information recorded in process execution logs. The methods are mostly rediscovering and recommending the execution order of process steps without facing possible coherency with other perspectives of business processes, e.g., incorporated data, agents performing the work and utilised tools [5]. The reasons, e.g., for a given execution order, possibly influenced by various perspectives of recorded information, remain mostly undiscovered. We think that mining algorithms must be able to involve all the perspectives of a business process. On this way, it becomes possible to discover complex coherencies like, e.g., the actual performing agent of a process step affecting the type of data used. These factors are the real set screws influencing business process execution. Some hidden coherencies or patterns even cannot be discovered by interviews or observations. For this purpose, we propose a method to discover such so-called *cross-perspective* dependencies [6] in process execution logs and therefore strive for a *genotypic* analysis, i.e., analysing the reasons for the things that happened, of recorded process data. The perspectives of a business process are mutually interacting in different shapes, e.g., document formats (*data* perspective) are influencing the tool (*operational* perspective) that performing agents (*organizational* perspective) have to use. In order to achieve a real discovery of coherencies, the approach at hand makes use of the traditional data mining technique of *association rule mining* [7-8]. Since this presents a fully developed data mining algorithm and in case the input dataset is specified well, the provided results are accurate. As association rule mining originates from another application area (typically market basket analysis), process execution logs have to be pre-processed and transformed in order to serve as an input for association rule mining. The resulting association rules can be used for online decision support and for guiding process participants through process execution.

Founded by „Europäischer Fonds für regionale Entwicklung (EFRE)“

II. General Approach and Preliminaries

Information systems typically log various kinds of information about process execution in a process execution log. A log consists of a set of traces whereat each trace is a sequence of events corresponding to a particular case, i. e., a single process instance [14]. Each recorded event refers to a single process step and has a timestamp recorded in the log. We demand for an existing log recording different perspectives of process execution. We recommend to record data based upon the different aspects of the perspective oriented process modelling (POPM) [5]:

Functional perspective: the perspective identifies a process step and defines its purpose. Also the composition of a process is determined by this perspective. Hence, the log should contain a common process identifier the corresponding event can be linked to.

Data perspective: the data (flow) perspective defines data used in a process and the flow of data between process steps. Therefore, the log should record documents or generally information that was used and produced by the current process step.

Operational perspective: the operational perspective specifies which operation is invoked in order to execute a process step. The log should contain applications or services that were used during performing the currently executed process step.

Organizational perspective: the perspective defines agents (users, roles) who are eligible and/or responsible to perform a process step. The log contains information about the process executor.

Behavioural perspective: the behavioural perspective is used to define dependencies between process steps (e.g., step B may only be executed after step A) that cannot be described by other perspectives (unexplained dependencies). The information in the log concerning this perspective is formed by the recorded timestamp of each event.

Table 1 shows a fragment of a process execution event log containing information about the described perspectives of a business process.

TABLE I. A fragment of a process execution event log.

#	PID	Case	Action	Agent	Data	Tool	Time
1	**A**	**1**	Start	Trainee	sales.odt	Office	…
2	**A**	**1**	Finish	Trainee	demands.odt	Office	
3	**A**	*2*	Start	Manager	sales.odt	Office	
4	**A**	*2*	Finish	Manager	demands.odt	Office	
5	**B**	**1**	Start	Head	demands.odt	Excel	
6	**B**	*2*	Start	Head	demands.odt	Excel	
7	**B**	**1**	Finish	Head	toDoList.xls	Excel	
8	**B**	*2*	Finish	Head	toDoList.xls	Excel	
…	**…**	…	…	…	…	…	

We propose a three phase approach to adapt association rule mining algorithm to analyse a log. Performing these phases, we provide a method to discover cross-perspective dependencies [6] in process logs and strive for a genotypic analysis, i.e., analysing the reasons for the things that happened, of recorded data instead of analysing phenotypes, i.e., best possible reproducing the things that happened. While traditional process mining approaches are discovering execution orders of process steps, they lack to comprise the different perspectives of process data. Including more aspects of recorded data in analysis methods, it is possible to enhance extracted models with information regarding coherency between the process perspectives.

III. Preprocessing the Log to Instance Traces

An instance trace graph describes the execution order of process steps of a process (case). These trace graphs a very similar to graphs enacted in [12], however, we feature these graphs with context data of the organisational, data and the operational perspective. Every node has the following fields: process name, performing agent, input and output document and used tool. Note, that we consider only one item per perspective with respect to comprehensibility reasons. Every edge has a field *execution type* (parallel or sequence) describing how two processes are connected. First, we separate the recorded events according to their corresponding case. Therefore, we assemble a list for each case represented in the log and assign the events according to their case ids. We can now classify the relation between two (sub-) processes within one process case. The classification is based upon the event types of two succeeding events. We make the same assumptions as [12]. Consider two processes *A* and *B*. We deduce that two process are executed in parallel if process *A* is started before process *B* is started and completed before *B* is completed but after the start of *B*. This would result in the event sequence: *Start A*, *Start B*, *Finish A*, *Finish B*. Furthermore, the two processes are also executed in parallel if process *A* is started before process *B* is started and completed after process *B* is completed. The resulting event sequence would look like this: *Start A*, *Start B*, *Finish B*, *Finish A*. In addition to parallel execution, we mark direct sequential execution. Two processes *A* and *B* are executed in a direct sequence if process *B* is started directly after process *A* has been completed. The resulting event sequence therefore is: *Start A*, *Finish A*, *Start B*, *Finish B*. An instance trace of an event list is created as follows. We generate a graph by running through the case-specific event list. For every newly occurring process step *A* within the list, we create a new node *A* within a graph and assign the corresponding process context. For every direct sequential-relation of two process steps *A* and *B*, we add an edge of execution type "sequence" between the nodes of *A* and *B*. For every parallel execution between two process steps *A* and *B*, we add an edge of execution type "parallel" between the nodes of *A* and *B*. Fig. 2 shows four different instance trace graphs of a process based upon the log fragment of Table 1 (for space reasons, the table just shows the activities of two instances). Considering graph no. 1, case 1 had the execution trace *A*, *B*, *C*, *D*, containing only direct sequential-relations and no parallelism. Exemplarily, graph 1 additionally contains the information that the agent "Trainee" executed process step *A* by using the document "sales.odt", produced the document "demands.odt" and was supported by the "Office" tool. The other three graphs can be interpreted in the same way. These graphs form the basis for the transformed input dataset in Chapter 4.2.

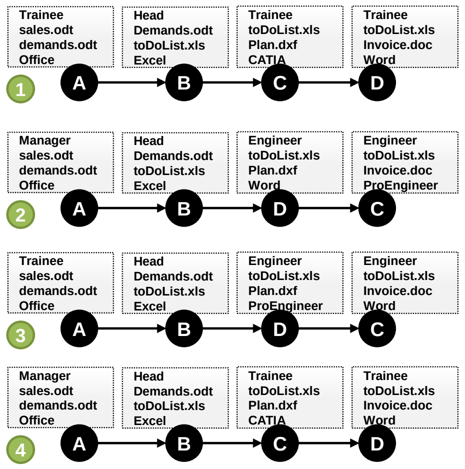

Figure 2. Instance trace graphs based on an exemplary process log.

IV. Transformation of Instance Trace Graphs to Dataset for Association Rule Mining

In this chapter, we will describe how the instance trace graphs are transformed into an input dataset for association rule mining.

A. Required Dataset for Association Rule Mining Algorithm

Association rule mining finds all rules in a given dataset that satisfy some minimum support and minimum confidence constraints [7-8]. For association rule mining, the target of mining *is not predetermined*. This is the fundamental difference to our previous approach of discovering cross-perspective dependencies in process execution logs [6]. Association rules are extracted from a set of instance traces using the *Apriori algorithm*. Traditionally, the Apriori algorithm [7] extracts interesting associations among a large set of data. This algorithm shows attribute value conditions that occur frequently together in a given dataset. The required dataset is a normal table, which consists of *n* columns described by *n* distinct attributes. An attribute can be a categorical (or discrete) or a continuous (or numeric) attribute. We treat all the attributes uniformly, like in [9].

B. Transformation of Instance Traces to the Required Dataset

The transformation of instance traces to a dataset for association rule mining can be defined as follows: let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of process steps containing process execution information (process name, performing agent, input and output document and used tool) and let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of instance trace graphs. Each instance trace in I contains a subset of the process steps in P as its nodes and a set of edges E with $e \in E$ in the form $P_x \rightarrow P_y$ (direct sequential execution) or $P_x \leftrightarrow P_y$ (parallel execution). One single *itemset* is assembled for every edge of each $i \in I$. Therefore, an itemset consists of seven attributes: performing agent of P_x, performing agent of P_y, the output document of P_x, the input document of P_y, the utilised tool of P_x, the utilised tool of P_y and finally the execution order type describing how the two processes P_x and P_y are connected (sequential or concurrent). We enable the Apriori algorithm to analyse dependencies between the perspectives regarding a single process step (e.g., how is the performing agent influencing an utilised tool) as well as two successional process steps (e.g., extracting causes for the given execution order of process steps). Regarding the instance traces of Fig. 2, the resulting dataset for association rule mining is shown in Table 2. The example shows four instance traces with three edges in each case. According to the transformation described above, this would result in a dataset of 12 itemsets.

TABLE II. Process log transformed to dataset for association rule mining.

Agent(x)	Agent(y)	Doc_Out(x)	Doc_In(y)	Tool(x)	Tool(y)	Order
Trainee	Head	demands.odt	demands.odt	Office	Excel	A → B
Head	Trainee	toDoList.xls	toDoList.xls	Office	CATIA	B → C
Trainee	Trainee	plan.dxf	toDoList.xls	CATIA	Word	C → D
Manager	Head	demands.odt	demands.odt	Office	Excel	A → B
Head	Engineer	toDoList.xls	demands.odt	Excel	Word	B → D
Engineer	Engineer	invoice.doc	demands.odt	Word	ProE	D → C
Trainee	Head	demands.odt	demands.odt	Office	Excel	A → B
Head	Engineer	toDoList.xls	demands.odt	Excel	ProE	B → D
Engineer	Engineer	plan.dxf	demands.odt	ProE	Word	D → C
Manager	Head	demands.odt	demands.odt	Office	Office	A → B
Head	Trainee	toDoList.xls	toDoList.xls	Office	CATIA	B → C
Trainee	Trainee	plan.dxf	toDoList.xls	CATIA	Word	C → D

V. Using Apriori Algorithm to Discover Patterns

We use the Apriori algorithm to extract a set of association rules from process execution information that was transformed to an appropriate form of an input dataset (Table 2). An association rule is defined as an implication of the form $X \Rightarrow Y$. X and Y consist of a set of values with respect to seven attributes. In order to select interesting rules from the set of all possible rules, this technique needs two parameters: the *Support Threshold* (1) and the *Confidence Threshold* (2). The support of an itemset is defined as the proportion of itemsets in the collection which contain the elements of the currently observed itemset:

$$Support(X \Rightarrow Y) = \frac{\#Itemsets\ containing\ X\ and\ Y}{\#Itemsets} \qquad (1)$$

The confidence threshold regarding a rule is defined as the conditional probability that Y is true when X is known to be true for a random instance:

$$Confidence(X \Rightarrow Y) = \frac{Support(X \Rightarrow Y)}{Support(X)} \qquad (2)$$

Also, confidence can be interpreted as the probability of finding the RHS (right-hand-side) of the rule in an itemset under the condition that this itemset also contains the LHS (left-hand-side) [9]. The Apriori algorithm must provide the minimum support threshold (*minSupp*) and the minimum confidence threshold (*minConf*) to verify the itemset. If the occurrence frequency of the itemset is greater than or equal to *minSupp*, an itemset satisfies the *minSupp*. If an itemset satisfies the *minSupp*, it is a frequent itemset. Rules that satisfy both a *minSupp* and a *minConf* are called strong. The Apriori algorithm can be explained following [9]. Therefore, we define:

C_k is a candidate *itemset* of size k

L_k is a frequent *itemset* of size k

The main steps of the iteration are:

1) Find frequent set L_{k-1}

2) Join step:

3) C_k is generated by joining L_{k-1} with itself (Cartesian product $L_{k-1} \times L_{k-1}$)

4) Prune step (apriori property):

Any $(k-1)$ size *itemset* that is not frequent cannot be a subset of a frequent k size *itemset*, so it should be discarded

5) Frequent set L_k has been achieved

We used the WEKA Data Mining Framework [13] which implements a variety of data mining algorithms, including the described Apriori algorithm. Our example provides the *minSupp* as 20% and the *minConf* as 100%. Using a confidence threshold of 100% has the meaning that in every case the right-hand-side was found, also the left-hand side was found. The confidence threshold could also be set less than 100%. This has the consequence, that a pattern could even be discovered in case that it was biased due to exceptions during process execution. Considering the complete dataset of Table 2, the Apriori algorithm of WEKA discovered a variety of association rules across all the perspectives. Exemplarily, we point out and describe three interesting patterns, discovered by the algorithm. Note, that of course many more rules have been discovered.

$$Doc\_Output(x) = demands.odt \wedge Doc\_Input(y) = demands.odt \Rightarrow ExecutionOrder = A \rightarrow B \qquad (3)$$

The extracted association rule (3) says: if the produced document of the previously completed process has been “demands.odt” and this document has been utilised by the directly following process, then the concerning process execution order has always been “B directly started after A”. In fact, this rule discovers a data flow between the process steps A and B, resulting in a direct sequential execution order. Here, the data perspective influenced the resulting execution order.

$$Agent(x) = Trainee \wedge Doc\_Output(x) = plan.dxf \Rightarrow Tool(x) = CATIA \qquad (4)$$

In case that the performing agent of a process step is a Trainee and the document that should be produced during a process step is “plan.dxf”, rule (5) discovered that the tool to be used has always been the CATIA program. Note, that the document has obviously not always been produced with the CATIA program but only in case of a Trainee performing the process. Here, even two perspectives (organisational and data perspective) are influencing the operational perspective.

$$Agent(y) = Trainee \wedge ExecutionOrder = C \rightarrow D \Rightarrow Doc\_Input(y) = toDoList.xls \qquad (5)$$

Rule (5) says: if a Trainee wants to perform a process step D and the predecessor process step was C, then the agent should use the “toDoList.xls” document to perform the process. Here, the rules extracts that a Trainee has to use a todo-list in case of performing the process step D after C. Hence, the organisational perspective influences the data perspective, limited to the performed process execution order D after C.

VI. Application: Guidance through Process

Here, we will describe the usage of extracted rules for guiding process participants through process execution. Therefore, we present the Process Observation project [14] (PO). Here, process execution information is manually entered and finally recorded with the help of the PO logging interface. Process participants provide information about the process they are currently performing. On this way, a complete process execution log is assembled. During the whole process execution, this log is consistently analysed by process mining methods and process models are extracted. Hence, the PO interface is able to show recommendations of how to continue process execution by traversing the extracted process model. Another possibility to guide users is to show recommendations based on association rules discovered by the approach at hand. Therefore, the assembled log of the PO is *periodically* transformed to an input dataset for association rule mining. Subsequently, the Apriori algorithm is applied to this dataset. The algorithm finally extracts a set of cross-perspective association rules. These rules could be used as follows: consider the extracted rules (3) – (5). A "Trainee" employee just completed a process step C. Subsequently, he wants to perform a process step "D" and enters this name in the corresponding field of the PO interface. Being quite un-sure which document to use for performing step D, he asks the PO for a dynamic recommendation. The available information regarding the previously completed process step ("C") as well as the current performed process step ("D", Agent = "Trainee") is sent to the PO. Here, the rules are explored regarding the received information. If the information satisfies the complete left-hand side of a special rule, the right-hand side is sent back to the user as a recommendation. The Trainee gets the recommendation to use the "toDo-List.xls" in order to perform the process, according to rule (5). Everything is just a recommendation based on coherencies that can be found in the recorded process log. Users always have the possibility to deviate from the recommended data.

VII. Related Work

Starting point for process mining is a process execution log. In many cases, the basis for analysis is pre-processing the available log [4]. In this paper, the pre-processing is carried out by converting the log information to so-called instance trace graphs. These graphs are similar to the graphs of [12], however, we feature them with further context data of process execution. There are already several algorithms and even complete tools, like the ProM Framework [10], that aim at discovering and generating process models automatically. During the last decade, several algorithms have been developed, focusing different perspectives of process execution data [2]. In contrast to the approach at hand, most of the methods are analysing one single perspective without facing possible coherencies with other perspectives of business processes. We already introduced an approach for cross-perspective mining of process execution logs in [6]. However, this algorithm has an important drawback: the method needs a set of rules as an input. In order to achieve a real discovery of association rules, the approach at hand makes use of the traditional data mining technique of association rule mining [7-8], implemented by the Apriori algorithm within the WEKA framework [13]. Note, that this algorithm has rarely been used in order to analyse business processes. In [9] the authors used the Apriori method to extract rules, revealing which process steps are executed together in certain process instances, i.e., existence patterns. Here, coherency to process context information, i.e., coherency between all the business process perspectives, is again neglected.

VIII. Conclusion, Limitations and Outlook

In this paper, we proposed an approach to discover cross-perspective dependencies from process execution logs. The approach allows for a genotypic analysis, i.e., analysing the reasons for the things that happened, of recorded process execution data. In order to discover cross-perspective coherencies, we made use of the traditional data mining technique of association rule mining. Here, we adapted a process execution log to serve as an input for the well-known Apriori algorithm. We described the application of the resulting association rules for online decision support and for guiding process participants through process execution. However, we clearly have to state that the Apriori algorithm potentially discovers many rules. Therefore, in some cases it could be quite difficult to get a clear overview over the extracted rules with the naked eye. Nevertheless, many discovered rules are quite accurate for the described application scenario of guiding users through process execution. Here, a great number of extracted coherencies means powerful support of future process execution through recommendations. Our future research activity will start with the integration of the described approach in our Process Observation (PO) project [14]. Here, the generated recommendations during process execution can finally by enriched by association rules. This should finally yield to an improvement of collaborative work through the recommendation of best-practice patterns.

Acknowledgment

The presented system is developed and used in the project "Kompetenzzentrum fuer praktisches Prozess- und Qualitätsmanagement", which is founded by "Europäischer Fonds für regionale Entwicklung (EFRE)".

References

[1] Van der Aalst, W., Pesic, M., Song, M.: Beyond Process Mining: From the Past to Present and Future. In: Pernici, B. (eds.) Advanced Information Systems Engineering. LNCS, vol. 6051, pp. 38-52. Springer, Berlin-Heidelberg (2010)

[2] Van der Aalst, W.: Process Mining: Discovery, Conformance, and Enhancement of Business Processes, Springer, Berlin-Heidelberg, DOI 10.1007/978-3-642-19345-3, ISBN: 978-3-642-19344-6. (2011)

[3] Van der Aalst, W., Reijers,H., Weijters, A., Van Dongen, B., De Medeiros, A., Songa, M., Verbeek, H.: Business process mining: An industrial application. In: Information Systems, vol. 32, Issue 5, pp. 713-732. (2007)

[4] Van der Aalst, W., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. In: IEEE Transactions on Knowledge and Data Engineering, vol.16, no. 9, pp. 1128-1142. DOI: 10.1109/TKDE.2004.47 (2004)

[5] Jablonski, S., Bußler, C.: Workflow Management: Modeling Concepts, Architecture and Implementation. Thomson, London. (1996)

[6] Schönig, S., Günther, G., Zeising, M., Jablonski, S.: Discovering Cross-Perspective Semantic Definitions from Process Execution Logs. In: Proceedings of the Second International Conference on Business Intelligence and Technology (BUSTECH 2012), Nice, France. ISBN: 978-1-61208-223-3. (July 2012)

[7] Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the 1993 ACM SIGMOD international conference on Management of data, ACM, New York, USA. ISBN: 0-89791-592-5, DOI: 10.1145/170035.170072 (1993)

[8] Zhang, C., Zhang, S.: Association rule mining: models and algorithms. Springer, Berlin-Heidelberg, ISBN: 3-540-43533-6. (2002)

[9] Polpinij, J., Ghose, A., Dam, H.: Business Rules Discovery from Process Design Repositories. In: 6th World Congress on Services, pp. 614-620, DOI: 10.1109/SERVICES.2010.73. (2010)

[10] Van Dongen, D., De Medeiros, A., Verbeek, H., Weijters, A., Van der Aalst, W.: The ProM framework: A new era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) Applications and Theory of Petri Nets, LNCS, vol. 3536, pp. 444-454. (2005)

[11] Rozinat, A., Van der Aalst, W.: Decision mining in business processes. In: BPM Center Report BPM-06-10, BPMcenter.org. (2006)

[12] Pinter, S., Golani, M.: Discovering workflow models from activities' lifespans. In: Computers in Industry, vol. 53, Issue 3, pp. 283-296. (2004)

[13] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.: The WEKA data mining software: an update. In: ACM SIGKDD Explorations Newsletter, vol. 11, Issue 1, pp. 10-18, ACM, New York, USA. DOI: 10.1145/1656274.1656278. (2009)

[14] Schönig, S., Günther, C., Jablonski, S.: Process Discovery and Guidance Applications of Manually Generated Logs. In: Proceedings of the Seventh International Conference on Internet Monitoring and Protection (ICIMP 2012), Stuttgart, Germany. ISBN: 978-1-61208-201-1. (2012)