Titolo da decidere

authors

date

Abstract

This is a very short abstract.

1 Introduction

Process mining is a recent discipline that includes tecniques for process analysis and discovery. Today, many Information Systems that integrate the concept of business process, provide registration of events that occur during the execution of the process activities. Starting from recorded executions related on the same process, the goal of the existing process mining techniques is extracting data to discover automatically a process model or, if the model is already exists, to check the executions in term of conformance and performance.

The idea that ispires this work consists in developing un approach for exploiting the huge ammount of data recorded at the process activities by information systems. More precisely, our purpose is to find pattern on data that could influence the process behaviour. For this reason, the work presented in this article is based on some concepts originated to Machine Learning and Data Mining disciplines. Should be noted that using data mining for similar aims has already been explored by Van der Aalst in [], the idea here consists in discovering how data attributes may influence the routing of case. Instead, in this contribution the intention is to present an approach based on data mining techniques also, but for discovering data dependencies on the process conformance and performance. Finally, an another goal that we propose is providing a predective model to detect the conformance result and some performance metrics for new process executions.

In front of a very large number of process instances, all recorded in event log, it is essential to have an automatic tool for extracting usefull information for the analysis of the real behaviour of the process. In order to meet this requirement, in this article we present an approach based on classification and decision trees to identify rules that could influence the process in term of conformance to the existing model, and performance metrics.

The article is organized as follows. After an introduction to the different formalism used to carry out our purposes, the attention will be...

2 Example

To clarify what presented in this article the example illustrated in this section will be taken into exam whenever is needed.

Commercial organizations usually follow a specific procedure to manage orders received form various clients. In general this procedure represents a business process that includes many differents activities, each of theme can involves diffent departments of the organization. Let us consider a sempli-facation of this procedure. For us, the sale process begins with receving an order notification from a client, so the first process activity is simply called "Order". After the order receipt the sale process continues with some activities that can be done in a parallel way (since they do not present any dependencies). One of thes is the "FianancialCheck" activity during which the order is analysed from a financial point of view (fo example verifying the client situation and solvency). In parallel an activity called "WarehouseCheck" is performed. In this phase a check regarding the merchandise requested by the order is done, and in case of a shortage an external order is issued for a supplying, this is done through the activity "ExternalWarehouse". After the completion of the parallel activities, a synchronization is needed before the continuation of the sale procedure. In the semplification considered here the sale process terminates with the activity "Notify" in which a result regarding the acceptance of the order is comunicated to the client.

The are many differents formalisms for modeling business process, the most popolar are based on flowcharts because of their expressivness (BPMN fo instance). For our purposes, a process is represented by a Petri net which is a matematical model so rigorous and better suited to process mining anal-

ysis. It is worth noting that they exist some techniques that allow traduction of a model espressed in a flowchart to a model espressed with a Petri net, more information can be found in []. The petri net representing the sale process presented in this section is in figure 2.

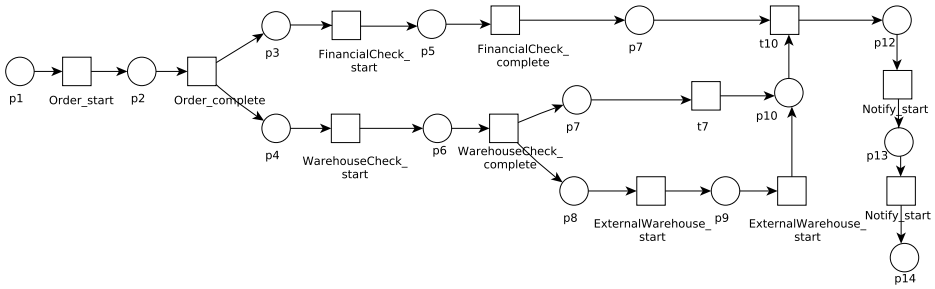

Figure 1: Petri net

3 Conformance and Performance Analysis based on Petri nets

The process analysis is performed by using the Petri net modelling the process and an event log recording the related executions (process instances). The basic building blocks of event logs are events. An event e can be seen as a pair $e = (a,t)$ representing an action a recorded and the corresonding timestamp t. Action and timestamp are denoted respectively by $\alpha(e)$ and $\phi(e)$. Events that belong to the same process are grouped into *traces*. Formally, a trace T is a finite sequence of events $T[1], ..., T[n]$, such that $\phi(T[i]) \leq \phi(T[i+1])$ for all $i \in [1, n)$. A log L is a set of traces, recording the activities performed by a system during a finite number of process executions. We assume here the following hypothesis:

- All traces are instances of the same process.
- For each action there exists a corresponding transition in the net that will be denoted, for simplicity, by the same name of the action.

The key algorithm expoloited to analyze the Petri net model with respect to the log is the *log repaly* algorithm. Given a Petri net model and an event log as intput to the algorithm, the output results can be used to check the

conformance of traces and to evaluate some performance metrics. For each trace in the log, log replay starts by placing one token in the start place of the net. For each event in the trace the corresponding transition is fires in an *non blockin way* and the marking of the net is updated. "Non blocking replay" means that if the log replay execution requires to fire an enabled transition, the missing tokens are created artificially. The output of the log replay of a trace can be represented as an orderd list R of pairs (tr, i), representing that the transition tr has been fired to mimic the event $T[i]$.

In general, there could be exist some transition in the net called *invisible*. This can happen if the transition models an internal choice that is not visible in the system, or it is used to implement a construct of a more abstract modelling language (BPMN or others). Usually, invisible transitions are considered to be lazy, i.e, when firing a visible transition t corrisponding to an event of the trace, only then the invisibile transition enabling t is fired. However, in this paper, we adopt an other way for handling these transitions and the firing is performed as soon as possible. This allows to carry out some performance measures as explained that are not possible with the lazy metohode as explained in [**?**](riferimento al paper Applying Process Analysis to the italian eGov Entreprise Architecture)

The result of log repaly can be used to evaluate conformance and performance of the Petri net model. Conformance problems can be discovered by analyzing the tokens that have been artificially created *(the missing tokens)* and the tokens tha were not consumed *(the remaining tokens)*.

To clarify the conformance analysis technique we exploit the Petri net presented in section 2 with reference to tow traces , T and T', presented respectively in figures 3 and 3. The log replay execution of the trace T, which is compliant with the Petri net, terminates with a marking containing one token in the end place $\{p14 \rightarrow 1\}$, and returns the sequence:

$$\begin{aligned} R = \{&(Order\_start, 1), (Order\_complete, 2), (WarehouseCheck\_start, 3), \\ &(FinancialCheck\_start, 4), (WarehouseCheck\_complete, 5), \\ &(FinancialCheck\_complete, 6), (t10, 6), (Notify\_start, 7), (Notify\_complete, 8)\} \end{aligned} \tag{1}$$

The log replay execution of the trace T' terminates with remaing tokens $\{p8 \rightarrow 1, p14 \rightarrow 1\}$ and a missing token $\{p10 \rightarrow 1\}$. The missing token is created artificially and this fact records a wrong execution of the event

Order #1 start 02.12.2012 9:30:50	Order #1 complete 02.12.2012 10:15:00	WarhouseCheck #1 start 02.12.2011 10:35:25	FinancialCheck #1 start 02.12.2012 10:40:20	FinancialCheck #1 complete 02.12.2012 12:00:20	WarhouseCheck #1 complete 02.12.2012 12:40:20	Notify #1 start 02.12.2012 12:55:20	Notify #1 complete 02.12.2012 13:00:10

Figure 2: Conforme log example

$Notify\_start$. In fact, notice that in the trace T' this event is executed before the termination of the activity "FinancialCheck", and this is interpreted as a non conformance to the process model. The sequence returned by log replay is:

$$R' = \{(Order\_start, 1), (Order\_complete, 2), (WarehouseCheck\_start), (FinancialCheck\_start, 3), (WarehouseCheck\_start, 4), (t7, 5), (t10, 6) (Notify\_start, 6), (Notify\_complete, 7)\} \quad (2)$$

Order #1 start 02.12.2012 9:30:50	Order #1 complete 02.12.2012 10:15:00	WarhouseCheck #1 start 02.12.2011 10:35:25	FinancialCheck #1 start 02.12.2012 10:40:20	WarhouseCheck #1 complete 02.12.2012 12:40:20	Notify #1 start 02.12.2012 12:55:20	Notify #1 complete 02.12.2012 13:00:10

Figure 3: Non conforme log example

Since the logs contain timestamps, the log replay can be used to compute performance measures of the process. The idea is to calculate the time interval between production and consumption of tokens in each place. This tecniques can be applied only to traces that do not present missing token during the replay, because such tokens cannot have time information. During the log replay the following metrics can be computed for catch trace and each place:

- sejour time (tsj) : the time interval between arrival and depature of tokens;
- synchronization time (tsc): the time interval between arrival of a token in the place and enabling of a transition in the post-set of the place;
- waiting time (tw): the time interval between enabling of a transition in the post-set of the place and token departure (thus $tsj = tsc + tw$).

To clarify the metrics evaluated by this tecnique, as done with conformance analysis, we exploit the Petri net presented in section 2 and the

trace T presented in figure 3. The replay starts with firing the transition $Order\_start$ at time $0s$, thus a token arrives at $p2$ at time $0s$. After $45min$ the token in $p2$ is consumed and the transition $Order\_complete$ is fired, so $tw(p2) = 45min$, and a token arrives at $p3$ and $p4$. At the time $1h : 5min : 25s$ the transition $WarhouseCheck\_start$ is fired, thus $tw(p4) = 20min + 25sec$ and a token is produced at $p6$. At the time $1h : 10min : 20s$ the transition $FinancialCheck\_start$ is fired and a token is produced at $p5$, so $tw(p3) = 25min + 20s$. After $2h : 30min : 20s$ from the replay start the transition $Financial\_complete$ is fired, hence we have $tw(p5) = 1h + 20min$ and a token is produced at $p11$. At time $3h : 10min : 20s$ the firing of the transition $WarehouseCheck\_complete$ is executed, so $tw(p6) = 2h + 4min + 55s$ and token is produced at $p7$ and $p8$. At the same time the invisible transition $t10$ is fired and a token is produced at $p10$. At this point the transition $t10$ is enabled, so firing it is now possible and a token is produced at $p12$. Notice that $tsc(p11) = 40min, tsc(p10) = 0s$. At the time $3h : 25min : 20s$ the transition $Notify\_start$ is fired and a token is produced in $p13$, after $4min + 4sec$ the trasition $Notify\_complete$ is fired and $tw(p13) = 4min + 40s$.

An important performance measure is the activity execution time. This can be deducted from the waiting time in the places between the start and the activity complete transitions. For example, time execution for "FinancialCheck" is given by the waiting time computed for the place $p5 : tw(p5) = 1h + 20min$.

It is worth noting that the synchronization time is greater than zero only for places which contain on their post-set transitions depending from other places. For the Petri net in figure 2, the only places which can have a synchronization time not null are $p10$ and $p7$. For instance, the sycronization time is greater than zero for $p10$ in the trace in figure 3. That means that, for this particular instance of the process, the branch with the occurance of the activity "FinancialCheck" is faster than the branch with activities relating the warehouse.

4 An approach based on classification for conformance analysis

Employing data mining techniques for process analysis was already experimented by Van Deer Aalst in [**?**]. In this article [], the idea consists in discovering how data attributes may influence the routing of case. The

motivation that encourages this work consists in the presence of a huge ammount of data recorded in the event logs in the form of attributes that could contain implicit information. In this contribution, the same methodology is adopted to offer an additional tool for process analysts.

With the approach presented here, the goal proposed is discovering patterns or rules in the event logs data in corrispondence of which conformance errors, discussed in section 3, occur. The goal is to conduct an investigation into data to discover the cause of conformance anomalies. Given the enormous quantity of data, the analysis is carried out with the classification, one of data mining techniques. In a classification problem, data is represented by a collection of records (called instances too), each of them is caracterized by a tuple $(\mathbf{x}, y)$, where $\mathbf{x}$ is an attribute set, while y is a special attribute called "target attribute" denoting the belowing class of the record. With this technique, the idea is learning a classification model that maps each attribute set x to one of the predefined class labels y. See [] for more information about classification.

Analysing data arising from event logs can be transformed into a classification problem. The data set can be derived from the process data, in particular, each instance data determines a record of the data set. For the process presented in section 2, a record is charaterized by the following attributes set: an instance identifier, a client identifier, the client typology with two possible values: new client and consolidated client, the sales manager responsabile for the order, the chief financial officier name who conducts the financial evaluation activities, the warehouseman name for the warehouse check activities, a supplier name in case of an external warehouse acitivity, and finally the result comunicated to the client for the order issued. The conformance result for the process instance provides an additional and important data which takes the role of the target attribute. All these attributes are discrete and contribute to formulate the data set presented in table 1 of the classification problem.

OrdIde	CltIde	CltType	SalMan	FinOff	WrhsMan	SplIde	OrdResut	Conf
1	20	consolidate	Marc	Alexander	Alex	15	positive	yes
2	700	new	Johann	Mario	Alex	04	positive	yes
3	10	consolidate	Mary	Robert	Alex	15	negative	yes
...	...	...	...	...	...	...	....	....
...	...	...	...	...	...	...	....	...

Table 1: Data set

Many classification models could be used for a classification problem. For our purpose we choose the decision tree model because of its simplicity and diffusion. In a decision tree, each leaf node is assigned a class label (a values of the attribute target). The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics. In principle, there are exponentially many decision trees that can be constructed from a given set of attributes. While some of the trees are more accurate than others, finding the optimal tree is computationally infeasible because of the exponential size of the search space. However, efficient algoritms have been developed to induce a reasonable accurate decision tree in a a reasonable ammount of time. These algorithm usually employ a greedy strategy that grows a decision tree by making a series of decisions about which attribute to use for partioning the data. One is Hunt's algorithm, which is the basic of many existing desicion tree induction algorithms, including $C4.5$ used for our purpose.

Using existing data mining tools for classification, it is possible to build a decistion tree as a classification model for the example presented here, based on data produced artificially just for examplary purposes: from the event log recorded, the attributes that characterized the process are extracted, and all the resources needed for the classification algorithm employed are produced. The decision tree resulted is presented in figure 4.

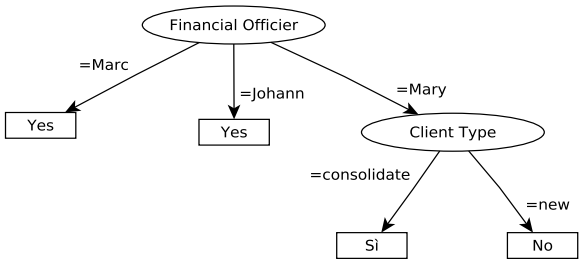

Figure 4: decision tree

The decision tree returned descibes a data pattern in corrispondence of which a process instance could present conformance errors: order managed by the sales manager "Mary" and received from consolidate clients of the organization may not respect the standard sales procedure. However, to get information most significant for our analysis, it is usefull to relate what is

noticed by the decision tree and the log replay results. In figure 4 is presented a Petri net that resumes the conformance analysis conducted on instances recorded in the event log taken in exam. Arcs are labled by the number of activations done, red colored places presented missing or remaning tokens and colored transitions notice a wrong execution.

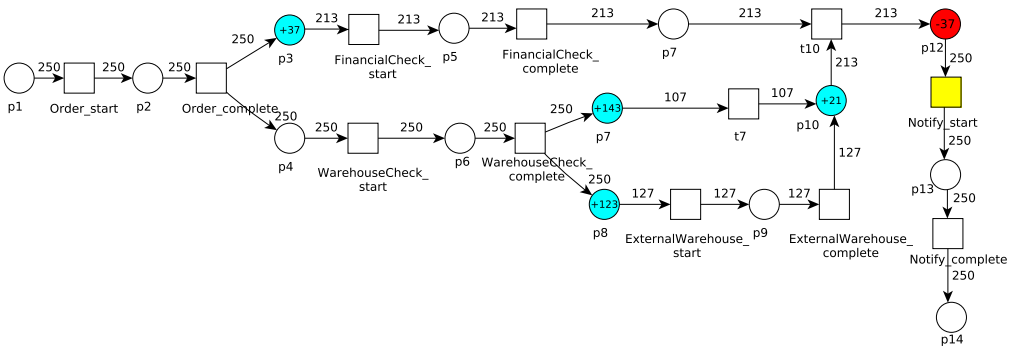

Figure 5: Petri net

The Petri net reporting the log replay results notice that there are 37 of 250 process executions with conformance erros, that is proved by the 37 missing tokens on the place $p12$ and a wrong execution of the transition "Notify_start". In order to give an interpretation for this results, it must be taken into account the non blocking behaviour of the log replay. In this case, the 37 missing tokens in $p12$ notice that during the replay the algorithm needs to mimic the event "Notify_start" but the transition associated to the event is not enabled in the marking reached by the net, so the algorithm creates artificial tokens for completing the trace repaly. This situation is possible only if the invisible transition $t10$ is not enabled. Now, if $t10$ is not enabled that means that there are no tokens in the place $p10$ and $p7$. But notice that there are 21 remaining tokens in the place $p10$, so the transition $t10$ is not enabled for those 37 instances because the place $p7$ does not present tokens. Moreover, the Petri net in figure highlights 37 remaing tokens in the place $p3$, so we can deduce that in those instances financial activities were not performed. From this facts, we can conclude that the conformance problems presented in 37 instances are due to a missing execution of the Financial Activities check. On the other hand, the decision tree reports that the instances with conformance problems are made by consolidate clients. In this case, it might be reasonable that, for the orders made by

the consolidate clients of the organization, could not be necessary a financial valutation of their situation.

Analysing the log replay results tanking into consideration patterns discoverd with classification technique leads to discover new cases to include in the process model by an extension. In other cases, if the process model is a representation of a fixed procedure to be respected, what discovered with such analysis can be used to take corrective measures. In fact, through a combination of the log replay results and the data patterns rules it may be possible to discover conformance errors cause, and this is fundamental to bring the right corrections. For instance, in the example presented here, if the model process represents a sales procedure that must be respected in any case, an appropriate correction can be taken to correct the wrong behaviour of the staff about consolidate clients.

COMMENTO: manca un pezzo che spiega come il classificatore puo' essere usato in senso predittivo aumentando la potenza del metodo. Mostrare a titolo di esempio che vantaggio ne possiamo trarre.

5 An approach based on classification for performance analysis

The approach presented in section 4 can be used also for performance analysis. Using classification tecnique for discovering how data attribute can influence process performance provides usefull information in analysing and optimazing the process services. For example, discovering data patterns in corrispondence of which some activities need more time for completion than others, helps in making decision about resources distribution to various process activities or in scheduling activities.

In addition to the execution time, classification can be used to discover information about more complex performance metrics such as the syncronization time. Processes with parallel activities and syncronizations can often present activities that represent a bottle neck, this leads to increase waiting time of some activities and consequently the completion time of the entire process. In this context, with the approach presented in this article, the objective is discovering rules in the process data that influence the syncronization time. The model builded by classification techinque can be used also in a predicting way, and this is certainly very important in order to optimize performance execution of the process.

With reference to the example taken in exam for conformance analysis, in this paragraph the approach based on classification is illustrated for the performance analysis.