DTMiner: A Tool for Decision Making Based on Historical Process Data

Josue Obregon, Aekyung Kim, and Jae-Yoon Jung

Dept. of Industrial and Management Systems Engineering, Kyung Hee University,
1 Seochen-dong, Giheung-gu, Yongin, Gyeonggi, Republic of Korea
{jobregon,akim1007,jyjung}@khu.ac.kr

Abstract. Process mining is a discipline that uses techniques to extract knowledge from event logs recorded by information systems in most companies these days. Among main perspectives of process mining, organizational and time perspectives focus on information about resources stored on the event logs and timing and frequency of the events, respectively. In this paper we introduce a method that combines organizational and time perspectives of process mining with a decision support tool called decision trees. The method takes the information of historical process data by means of an event log, generates a decision tree, annotates the decision tree with processing times, and recommends the best performer for a given running instance of the process. We finally illustrate the method through several experiments using a developed plug-in for the process mining framework ProM, first using synthetic data and then using a real-life event log.

Keywords: process mining tool, decision tree, decision making, recommendation.

1 Introduction

Data recorded by information systems are increasing in today's business environment allowing business analysis tools, which use this data to work, gain more and more value every day. One of these tools is process mining. The idea of process mining is to extract knowledge from the so-called event logs and discover, monitor and improve real processes. Process mining has three types of functions: discovery, conformance and enhancement. Discovery techniques take an event log as input and generate a process model as output using a plethora of notations like petri nets, causal networks, heuristic networks, and so on. Conformance techniques take an existing process model and compare with an event log in order to detect, locate, explain and measure deviations between the model and the actual execution of the process. Enhancement techniques extend or improve process models based on the information obtained the event log. Among different perspectives of process mining, in this paper we focus on two of them, the organizational perspective and the time perspective. Organizational perspective deals with the resource attributes of the event log (e.g., performers of activities), while time perspective considers timing and frequency of events (e.g. processing time of an activity) [1]. On the other hand, decision trees is a

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-3-319-02922-1_10

M. Song, M.T. Wynn, and J. Liu (Eds.): AP-BPM 2013, LNBIP 159, pp. 81–91, 2013.

decision-making tool that helps to clarify for management the choices, risks, objectives, monetary gains and information needs involved in an investment problem [2].

In this paper we take the method developed in [3] and verify its applicability by means of experiments using two kinds of data. The first experiment is conducted with synthetic data related with a repair process used in [4]. The second experiment is conducted with real-life data. Each experiment is accompanied with performance measures in order to evaluate its accuracy. The experiments are conducted using a developed plugin for the process mining framework ProM called *DTMiner*.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. Section 3 introduces the technique for constructing decision trees based on historical process data. Section 4 presents the implemented plug-in *DTMiner*. Section 5 presents the conducted experiments. Section 6 shows the results of the experiments and Section 7 discusses limitations, recommendations and conclusions of the paper.

2 Related Work

Process mining has proved its applicability in real life cases. In [5] a case study illustrating the practical application of process mining is presented. The authors pointed out that the case study showed that it is worthwhile to combine different mining perspectives to reach a richer understanding of the process. The method used in this paper also combines two perspectives of process mining, organizational and time perspectives.

Furthermore, in [6] a semi-automatic approach intended to reduce the number of manual staff assignment is described. Their approach applies a supervised machine learning algorithm to the process event log in order to learn the activities that each performer undertakes. Experiments on three enterprises' datasets were conducted and good overall prediction accuracy was achieved, reaching over 75%. In the technique used in this paper [3], process mining is not combined with machine learning algorithms, instead of that a decision support tool called decision trees is utilized and a simple algorithm for constructing the decision tree is used.

Another works [7, 8] also use machine learning approaches combined with process mining to achieve their results related with staff assignment and decision mining, respectively. In [7] they showed that the problem of deriving staff assignment rules using information from historical process data and organizational information can be interpreted as an inductive learning problem, therefore they used decision tree learning to derive meaningful staff assignment rules. In [8], a plug-in called *Decision Miner* that analyzes the choice constructs of a petri net process model in the context of the ProM framework was presented. Their approach converts every decision point within the process model into a classification problem, and then they solved that problem using decision tree learning.

It is important to remark that decision tree learning in the machine learning area is different from decision trees as a decision support tool. In [9] it is defined that decision tree learning (i.e., machine learning perspective) provides a powerful formalism for representing comprehensible and accurate classifiers, whereas in [2] it is stated that decision tree (i.e., decision analysis perspective) is a decision-making

tool that helps to clarify for management the choices, risks, objectives, monetary gains and information needs involved in an investment problem.

3 Performer Recommendation Using Process Mining

In this section we describe briefly the overall procedure used in [3]. The procedure of the proposed method is represented as shown in Fig. 1. In the first stage, a process model is discovered by process mining tools such as ProM, and a running case can also be observed. Then a decision tree is constructed based on the discovered process model and the event log. From the historical data, a key performance indicator (KPI) can be predicted, and information of performance prediction is projected onto the constructed decision trees. For example, the predicted completion time and cost of each pending tasks are annotated for performers.

In the second stage, a running case is matched with the constructed decision tree. To do that, the decision tree is simplified through filtering to reflect characteristics of process, and an observed running case is then matched to the decision tree. Finally, several subtrees can be extracted and merged by matching.

In the last stage, we finally recommend proper performers of each task. Performers are evaluated in terms of time and cost. We can recommend the best performers of scheduled tasks to improve a target KPI.

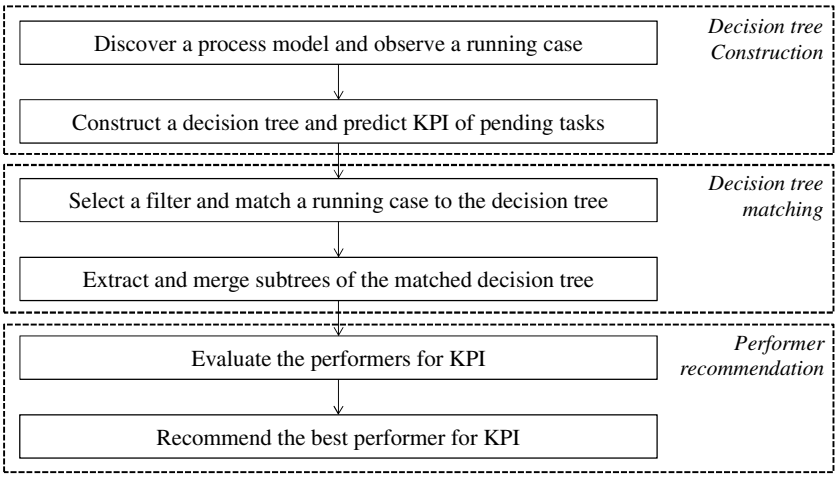


Fig. 1. Overview of performance recommendation based on historical data

4 DTMiner Plug-in

The technique presented in [3] was implemented as a plug-in for the ProM Framework. The ProM framework integrates the functionality of several existing process mining tools and provides additional process mining plug-ins [10, 11].

Furthermore ProM version 6 offers a new redesigned standard development environment, an enhanced architecture and the user interface that supports new developments on the process mining research area in a relatively easy way. ProM has five kinds of plug-ins, which implement different process mining related functions: mining, export, import, analysis and conversion. We center our attention on mining and analysis plug-ins. Mining plug-ins implement some mining algorithm, e.g., α-miner that constructs a Petri net based on some event log whereas analysis plug-ins implement some property analysis on some mining result[10].

The plug-in called *DTMiner* can be considered as a combination between mining and analysis plug-ins. The *DTMiner* plug-in constructs a decision tree based on an historical process data. In Fig. 2, a generated decision tree is depicted on the main screen of the plug-in interface. Decision nodes are colored with blue and have square shape meanwhile chance nodes are colored with green and have ellipse shape. Decision nodes represent tasks and chance nodes represent performers extracted from the event log. Node information is displayed when the mouse pointer is over the node and it varies depending on the type of it. If it is a decision node, remaining time and route are displayed and if it is a chance node, average time and frequency are displayed. The edge connection between chance nodes and decision nodes displays the average task time taken by the performer to finish the previous task. The underlying decision tree model used for the construction of the decision model stores all the information obtained from the event log. Each chance node is annotated with start and finish task times for each case of the performer that it represents as well as the case frequencies.

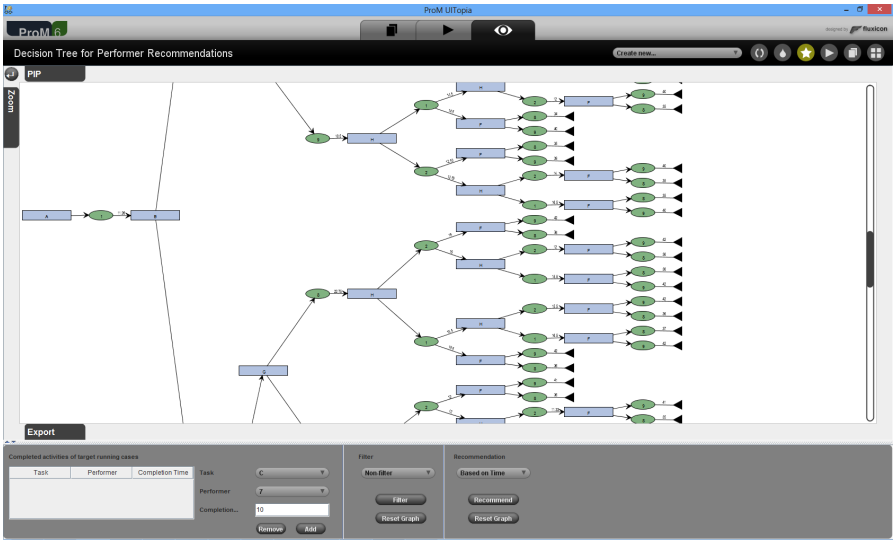


Fig. 2. Screenshot of *DTMiner* plug-in showing some results of test data

After loading the event log and generating the decision tree, one can analyze the resultant graph using the analysis section of the plug-in. The analysis section can be visualized at the bottom of the Fig. 3. It has three sections. In the first section, completed activities of target running cases can be added or deleted. In the second section, a filter to match the tree with the target running case can be selected. Finally on the third option a recommendation is given depending the parameter selected.

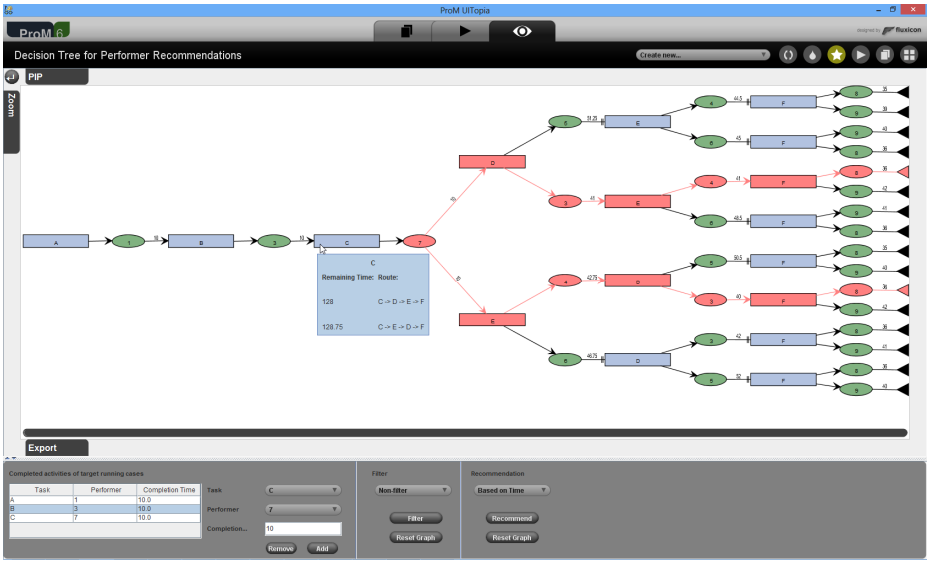


Fig. 3. Screenshot of *DTMiner* plug-in showing possible routes from the last task of the running instance and the recommended performers per route

Fig. 3 shows an example already filtered and analyzed. Using the filter non-filter the initial decision tree was pruned. After this, a recommendation based on remaining time is given. When the mouse pointer is over a task node, the possible routes from that point until the end are displayed. The remaining time for each route is also displayed beside the corresponding route. Recommended routes (i.e., nodes and arcs) are colored with red color whereas the arcs of the routes that are not recommended have two perpendicular lines indicating that are blocked.

In the next sections, the *DTMiner* plug-in is used as a proof-of-concept implementation over several event logs.

5 Experiments

In this section, we demonstrate the applicability of our approach using one synthetic event log obtained via ProM and a real-life log used in a case study. For the case study we analyzed a process in Dutch Financial Institute.

5.1 Synthetic Example

For the first experiment we use an event log about a process of repairing telephones in a company. In Fig. 4, we can see that the process starts by registering a telephone device sent by a customer. After registration, the telephone is analyzed and its defect is categorized. Once the problem is identified, the telephone is sent to the Repair department. The Repair department can fix simple defects and complex defects. Once a repair employee finishes working on a phone, this device is sent to the Quality Assurance department. Then the phone is analyzed by an employee to check if the defect was indeed fixed or not. If the defect is not repaired, the telephone is again sent to the Repair department. If the telephone is repaired correctly, the case is archived and the telephone is delivered to the customer.

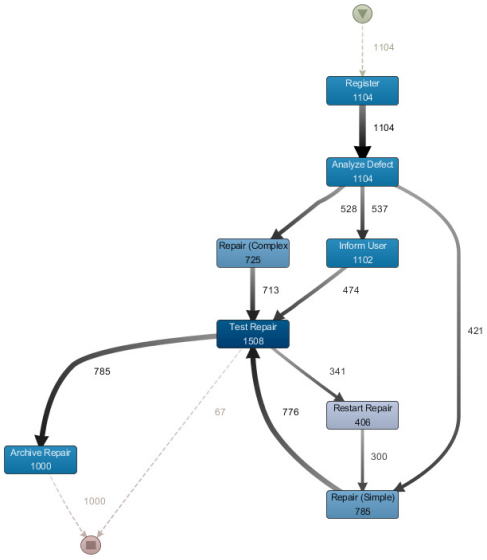


Fig. 4. Telephone repair process discovered by the improved fuzzy miner in Disco

In Fig. 4, it can be noted that 1,104 cases exist in the event log and begin with *Register* activity. Among those cases, just 1,000 cases have finished. We used cross-validation in our experiment. Cross-validation is the statistical practice of partitioning a sample data set into two subsets, training set and test set. Training set is used to analyze the data while testing set is used for validation. Because of the nature of the plug-in, in which every case should be tested by hand, our test set had a size of 10 cases and was selected randomly.

The experiment was conducted as follows. Take the real case from the test data, record the actual completion time and the performer. Use the plug-in and enter the first two activities as a running case and get the recommendation. After this, record the new recommended time and check if performers are different from the performers that actually executed the task on the test case. Repeat this for every case in the test data.

The results are summarized in Table 1. It is clear that the method works and always recommends the performer who has registered the shortest average time on the training data. This has a limitation that will be discussed later, about the fact that the recommended performer might be busy at the time when the running case is being executed.

Table 1. Summary of experiment results for synthetic data

Case	Total remaining time (min)	Recommended time (min)	Difference (min)	Number of performers changed
1	47	8.5	38.5	0
2	51	11.31	39.69	1
3	28	11.31	16.69	2
4	58	11.31	46.69	1
5	55	11.31	43.69	1
6	21	11.31	9.69	2
7	23	12.84	10.16	2
8	27	14.31	12.69	2
9	23	12.26	10.74	1
10	19	6.75	12.25	2

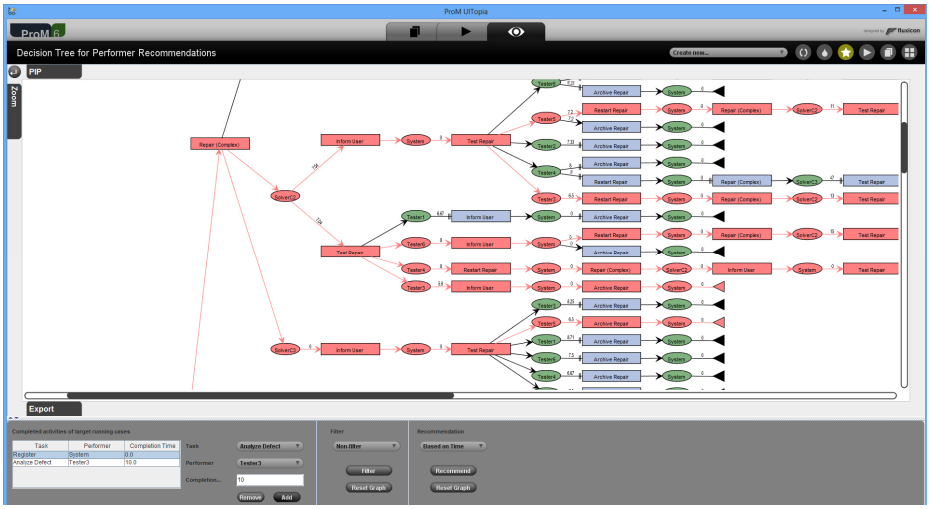


Fig. 5. Screenshot of *DTMiner* plug-in showing possible routes from the last task of a running instance and the recommended performers per route

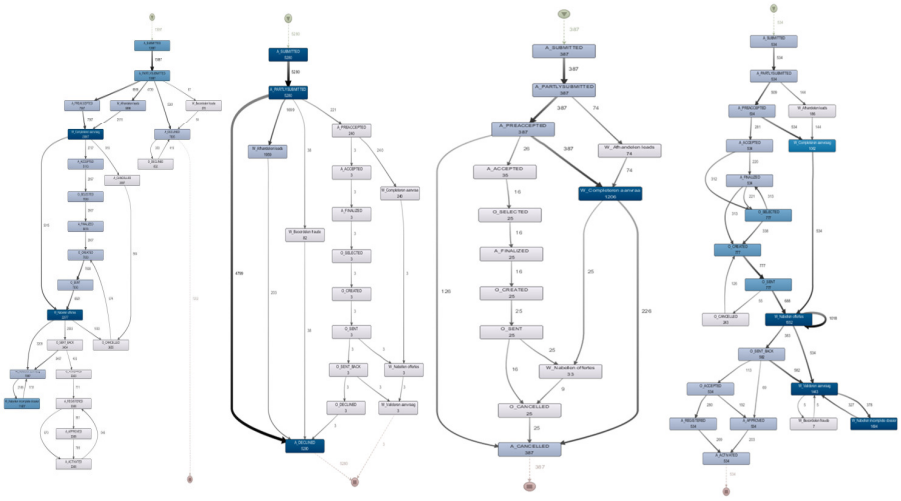
5.2 Case Study

We also evaluated the proposed approach using an event log from the Dutch Financial Institute. This log contains 13,087 cases and 262,200 events over a six month period from October 2011 to March 2012. The process represented in the event log is an

application process for a personal loan or overdraft within a global financing organization.

An incomplete case means unexpected case appearing because of extracting data from a particular period of time. Since information systems record events continuously, the log contains some cases which have not finished yet. To get rid of incomplete cases and provide some insight into the structure of the process, we used Disco which draws process models using the improved fuzzy algorithm. It also shows meaningful information such as variants, frequency, and duration and provides powerful filtering features. We found that the whole process can be split into three sub-processes by end events (i.e. *A_DECLINED, A_CANCELLED, A_ACTIVATED*). In the next subsection, we use the three groups split from whole cases, which contain 7,635, 2,807 and 2,246 cases, respectively.

There are some events in the log where the resource information is missing. For these reasons, before testing our approach the log needs to be preprocessed. We first removed all the cases which have at least one event with NULL resource information because they cannot be used for the performer recommendation method. Second, we used only cases whose sequence of activities is shared by at least 10 cases using variation filtering functionality of Disco. Moreover, we consolidated all the resources performed in automatic activities which have zero duration into a resource called 'Automatic'. Finally, we split the filtered log into three sub-processes. As a result the group that ends with *A_DECLINED* has 5,280 cases. The *A_CANCELLED* group has 1,024 cases and *A_ACTIVATED* group has 534 cases after filtering. Fig. 6 shows the process models of each group discovered by Disco.



(a) Whole Process (b) *A_DECLINED* (c) *A_CANCELLED* (d) *A_ACTIVATED*

Fig. 6. The process models discovered from Dutch Financial Institute's log

We present an example scenario with the log of the Dutch Financial Institute to describe how the proposed approach can be applied to performer allocation problems with the *DTMiner* plug-in. Using *DTMiner* with the example scenario, the historical process log of a business process was analyzed to construct the decision tree, which was used to recommend the best performers for an ongoing instance of the process.

The Dutch Financial Institute would want to reduce the lead time of their services to improve the quality of the customer loan service. They would also want to decrease the cost of their processes. For this reason, the purpose of this experiment is to recommend the best performer who allows the remaining time or the total labor cost to reduce for each next task.

As depicted in Fig. 1 the overall procedure of the proposed approach consists of three primary steps. Following these steps, we first constructed decision trees from the log using the plug-in. In this step, we used three sub logs and constructed decision trees separately.

In the second step, we assume that running case σ_1 = < (*A_SUBMITTED*, Automatic, 0, 0), (*A_PARTLYSUBMITTED*, Automatic, 0, 0), (*W_Afhandelen lead*, 10913, 3.9, 8), (A_PREACCEPTED, Automatic, 0, 0) > has been captured by the information system. Also, we suppose that a manager does not want to filter with previous performers, and he wants to obtain the recommendation of performers who can reduce the remaining time. We then set up the running case and filter options as shown in the bottom of Fig. 7.

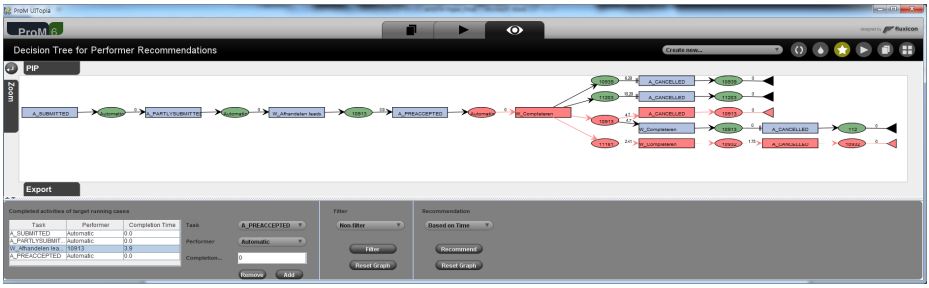


Fig. 7. Performer evaluation and recommendation for the sub-process that ends with '*A_CANCELLED*' with a running case σ_1

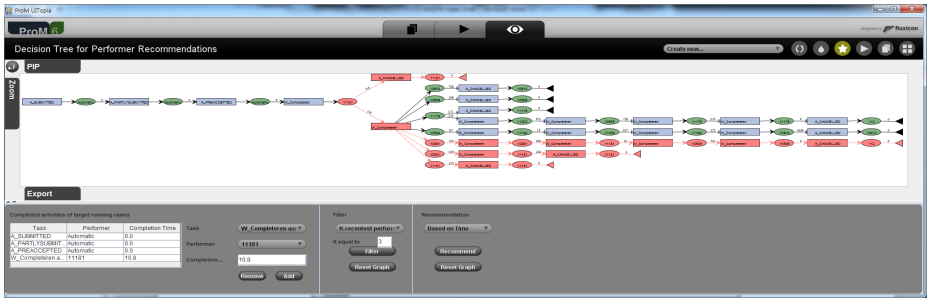


Fig. 8. Performer evaluation and recommendation for the sub-process that ends with '*A_CANCELLED*' with a running case σ_2

In the last step, information about the running case was matched with the decision tree and its subtrees were extracted and merged. Also, the predicted KPIs were updated. Finally, we evaluated performance and recommended the best performer for each next task by reducting inferior performers from leaf nodes. Fig. 7 shows the pruned decision tree and the best performer of each task in sub-process that ends with '*A_CANCELLED* ' for σ_1. After executing the running case σ_1, the pruned decision tree showed two possible traces with different excution probabilities as shown in Fig. 7. The first trace c_1 =< *A_SUBMITTED*, *A_PARTLYSUBMITTED*, *W_Afhandelen lead*, *A_PREACCEPTED*, *W_Completeren aanvraag*, *A_CANCELLED* > had an execution probability of 40% and the second trace c_2 = < *A_SUBMITTED*, *A_PARTLYSUBMITTED*, *W_Afhandelen lead*, *A_PREACCEPTED*, *W_Completeren aanvraag*, *W_Completeren aanvraag*, *A_CANCELLED* > had an execution probability of 60%. Based on this probabilities, we can recommend the best performer for task '*W_Completeren aanvraag*' is '10913' in c_1 of which the remaining time is 4.7 and is also the minimum remaining time. In the same way, the best performer for task '*W_Completeren aanvraag*' is '11181' and the best performer for task '*A_CANCELLED*' is '10932' in c_2. Also, Fig. 8 shows performer evaluation and recommendation for the sub-process that ends with '*A_CANCLLED*' when a running case σ_2 = < (*A_SUBMITTED*, Automatic, 0, 0), (*A_PARTLYSUBMITTED*, Automatic, 0, 0), (*A_PREACCEPTED*, Automatic, 0, 0), (*W_Completeren aanvraag*, 11181,10.8, 9) > is given and *3-recent filter* is selected.

6 Discussion and Conclusion

In this paper we introduced a tool for decision making based on historical process data. *DTMiner* is a combination between process mining principles and decision trees as a support decision tool. A decision tree is constructed based on an event log, and the decision tree is then annotated with activity processing times that later are used to recommend best performers based on some criteria. Two experiments were conducted with synthetic event log and real-life event log. Through the experiments we illustrated how the method can be applied to recommend good performers.

Some potential limitations still remain in the proposed approach. One limitation comes from the experimentation. Although the performance measures proved that the recommended performer can reduce the final completion time of the process instance, one cannot know if the recommender performer will be available at that moment. In the case is not available, the method should take in consideration waiting time until the performer is not busy anymore, or give an alternative recommended performer.

Another limitation is related with the notion of *completeness* in process mining [1]. One cannot assume to have seen all possibilities in the historical process data used to construct decision trees. If a running case is being evaluated and the sequence of activities was not recorded in the historical data, the method cannot give a recommendation because the branch which refers to the running case does not exist in the constructed decision tree. One way to overcome this limitation could be the use of process models when the decision tree is being constructed, adding possible behavior that actually does not occur but is still possible because of the process model.

Acknowledgments. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (Nos. 2012R1A1B4003505 and 2013R1A2A2A03014718).

References

1. van der Aalst, W.M.P.: Process mining: Discovery, conformance and enhancement of business processes. Springer, Heidelberg (2011)
2. Magee, J.F.: Decision trees for decision making. Harvard Bus. Rev. 42(4), 126–138 (1964)
3. Kim, A., Jung, J.-Y.: A process mining technique for performer recommendation using decision tree. In: Korean Institute of Industrial Engineers Conference (2012)
4. De Medeiros, A.K.A., Weijters, A.J.M.M.: ProM Framework Tutorial. TechnischeUniversiteit Eindhoven, The Netherlands (2009)
5. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., Alves de Medeiros, A.K., Song, M., Verbeek, H.M.W.: Business process mining: an industrial application. Inform. Syst. 32(5), 713–732 (2007)
6. Liu, Y., Wang, J., Yang, Y., Sun, J.: A semi-automatic approach for workflow staff assignment. Comput. Ind. 59(5), 463–476 (2008)
7. Ly, L.T., Rinderle, S., Dadam, P., Reichert, M.: Mining staff assignment rules from event-based data. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 177–190. Springer, Heidelberg (2006)
8. Rozinat, A., van der Aalst, W.M.P.: Decision mining in ProM. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 420–425. Springer, Heidelberg (2006)
9. Quinlan, J.R.: Decision trees and decision-making. IEEE T. Syst. Man Cyb. 20(2), 339–346 (1990)
10. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W(E.), Weijters, A.J.M.M.T., van der Aalst, W.M.P.: The ProM framework: Anew era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 444–454. Springer, Heidelberg (2005)
11. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: ProM 6: The process mining toolkit. In: BPM Demonstration Track, vol. 615, pp. 34–39 (2010)