# Process mining: Discovering direct successors in process logs

Laura Maruster[1], A.J.M.M.(Ton) Weijters[1], W.M.P.(Wil) van der Aalst[1], and Antal van den Bosch[2]

[1] Eindhoven University of Technology, 5600 MB Eindhoven, the Netherlands
{l.maruster, a.j.m.m.weijters, w.m.p.aalst}@tm.tue.nl
[2] Tilburg University, 5000 LE Tilburg, the Netherlands
antal.vdnbosch@kub.nl

**Abstract.** Workflow management technology requires the existence of explicit process models, i.e. a completely specified workflow design needs to be developed in order to enact a given workflow process. Such a workflow design is time consuming and often subjective and incomplete. We propose a learning method that uses the workflow log, which contains information about the process as it is actually being executed. In our method we will use a logistic regression model to discover the direct connections between events of a realistic not complete workflow log with noise. Experimental results are used to show the usefulness and limitations of the presented method.

## 1 Introduction

The managing of complex business processes calls for the development of powerful information systems, able to control and support the flow of work. These systems are called *Workflow Management Systems* (WfMS), where a WfMS is generally thought of as "a generic software tool, which allows for definition, execution, registration and control of workflows" [1]. Despite the workflow technology promise, many problems are encountered when applying it. One of the problems is that these systems require a workflow design, i.e. a designer has to construct a detailed model accurately describing the routing of work. The drawback of such an approach is that the workflow design requires a lot of effort from the workflow designers, workers and management, is time consuming and often subjective and incomplete.

Instead of hand-designing the workflow, we propose to collect the information related to that process and discover the underlying workflow from this information history. We assume that it is possible to record events such that (i) each event refers to a task, (ii) each event refers to a case and (iii) events are totally ordered. We call this information history the *workflow log*. We use the term *process mining* for the method of distilling a structured process description from a set of real executions.

To illustrate the idea of process mining, consider the workflow log from Table 1. In this example, there are seven cases that have been processed and twelve executed tasks. We can notice the following: for each case, the execution starts with task A and ends with task L, if C is executed, then E is executed. Also, sometimes we see task H and I after G and H before G. Using the information shown in Table 1, we can discover the process model shown in Figure 1. We represent the model using Petri nets [1]. The Petri net model starts with task A and finishes with task L. After executing A, either task B or task F can be executed. If task F is executed, tasks H and G can be executed in parallel.

**Table 1.** An example of a workflow log

| Case number | Executed tasks |
|---|---|
| Case 1 | A F G H I K L |
| Case 2 | A B C E J L |
| Case 3 | A F H G I K L |
| Case 4 | A F G I H K L |
| Case 5 | A B C E J L |
| Case 6 | A B D J L |
| Case 7 | A B C E J L |

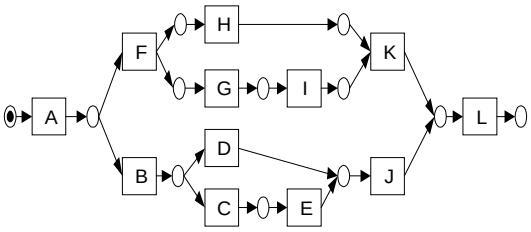


**Fig. 1.** A process model for the log shown in Table 1

A parallel execution of tasks H and G means that they can appear in any order.

The idea of discovering models from process logs was previously investigated in contexts such as software engineering processes and workflow management [2-9]. Cook and Wolf propose three methods for process discovery in case of software engineer processes: a finite state-machine method, a neural network and a Markov approach [3]. Their methods focus on sequential processes. Also, they have provided some specific metrics for detection of concurrent processes, like entropy, event type counts, periodicity and causality [4]. Herbst and Karagiannis used a hidden Markov model in the context of workflow management, in the case of sequential processes [6,8,9] and concurrent processes [7]. In the works mentioned, the focus was on identifying the dependency relations between events. In [10], a technique for discovering the underlying process from hospital data is presented, under the assumption that the workflow log does not contain any noisy data. A heuristic method that can handle noise is presented in [11]; however, in some situations, the used metric is not robust enough for discovering the complete process.

In this paper, the problem of process discovery from process logs is defined as: (i) for each task, find its direct successor task(s), (ii) in the presence of noise and (iii) when the log is incomplete. Knowing the direct successors, a Petri net model can be constructed, but we do not address this subject in the present paper, this issue is presented elsewhere [10, 11].

It is realistic to assume that workflow logs contain noise. Different situations can lead to noisy logs, like input errors or missing information (for example, in a hospital environment, a patient started a treatment into hospital X and continues it in the hospital Y; in the workflow log of hospital Y we cannot see the treatment activities that occurred in hospital X).

The novelty of the present approach resides in the fact that we use a global learning approach, namely we develop a logistic regression model and we find a threshold value that can be used to detect direct successors. As basic material, we use the “dependency/frequency tables”, as in [11]. In addition to the “causality metric” that indicates the strength of the causal relation between two events used in [11], we introduce two other metrics.

The content of this paper is organized as follows: in Section 2 we introduce the two new metrics that we use to determine the “direct successor” relationship and we recall the “causality metric” introduced in [11]. The data we use to develop the logistic regression model is presented in Section 3. Section 4 presents the description of the logistic regression model and two different performance test experiments are presented. The paper concludes with a discussion of limitations of the current method and addresses future research issues.

## 2 Succession and direct succession

In this section we discuss some issues relating the notion of succession and we define the concept of direct succession. Furthermore, we describe three succession metrics that we used to determine the direct succession relationship. At the end of this section we give an example of dependency/frequency table, with the corresponding values of the three metrics.

### 2.1 The succession and direct succession relations

Before introducing the definitions of succession and direct succession relations, we have to define formally the notion of workflow log and workflow trace.

**Definition 1**: *(Workflow trace, Workflow log)* Let $T$ be a set of tasks. $\theta \in T^*$ is a workflow trace and $W \subseteq T^*$ is a workflow log. We denote with $\#L$ the count of all traces $\theta$.
An example of a workflow log is given in Table 1. A workflow trace for case 1 is A F G H I K L. For the same workflow log from Table 1, #L = 7.

**Definition 2**: *(Succession relation)* Let $W$ be a workflow log over $T$, i.e. $W \subseteq T^*$. Let A, B$\in T^*$. Then:

- *B succeeds A* (notation $A >_W B$) if and only if there is a trace $\theta = t_1 t_2 \ldots t_{n-1}$ and $i \in \{1,\ldots, n-2\}$ such that $\theta \in W$ and $t_i = a$ and $t_{i+1} = b$.
  In the log from Table 1, $A >_W F$, $F >_W G$, $B >_W C$, $H >_W G$, etc.
- we denote $(A>B) = m$, $m \geq 0$, where $m$ is the number of cases for which the relation $A >_W B$ holds. For example, if for the log $W$, the relation $A >_W B$ holds 100 times and the relation $B >_W A$ holds only 10 times, then (A>B) = 100 and (B>A) = 10.

**Definition 3**: *(Direct succession relation)* Let $W$ be a workflow log over $T$, i.e. $W \subseteq T^*$ and A, B$\in T$. Then B *directly succeeds* A (notation $A \rightarrow_W B$) if either:

1. (A>B) > 0 and (B>A) = 0
   or
2. (A>B) > 0 and (B>A) > 0 and $((A>B) - (B>A) \geq \sigma)$, $\sigma > 0$.

Let us consider again the Petri net from Figure 1. A pair of two events can be in three possible situations and subsequently the relations between the events are:

a) if events C and E are in sequence, i.e. (C>E) > 0 and (E>C) = 0, then $C >_W E$ and $C \rightarrow_W E$.
b) if there is a choice between events B and F, i.e. (B>F) = 0 and (F>B) = 0, then $B \not> _W F$, $F \not> _W B$, $B \not\rightarrow _W F$, $F \not\rightarrow _W B$.
c) if events G and H are in parallel, i.e. (G>H) > 0 and (H>G) > 0, then $G >_W H$, $H >_W G$, $G \not\rightarrow _W H$, $H \not\rightarrow _W G$.

The first condition from Definition 3 says that if for a given workflow log $W$, only B succeeds A and A never succeeds B, then there is a direct succession between A and B. This will holds if there is no noise in $W$. However, if there is noise, we have to consider the second condition for direct succession, because both (A>B) > 0 and (B>A) > 0. The problem is to distinguish between a situation when (i) A and B are occurring in parallel and (ii) when A and B are really in a direct succession relation, but there is noise. In the rest of the paper we describe the methodology of finding the threshold value $\sigma$.

In order to find the threshold value $\sigma$, we use three metrics of the succession relation, which are described in the next subsection.

### 2.2 The local metric (LM), global metric (GM) and causality metric (CM)

**The local metric LM**. Considering tasks A and B, the local metric LM is expressing the tendency of succession relation by comparing the magnitude of (A>B) versus (B>A).
The idea of LM measure presented below is borrowed from statistics and it is used to calculate the confidence intervals for errors.

$$LM = P - 1.96\sqrt{\frac{P(1-P)}{N+1}},\ P = \frac{(A>B)}{N+1},\ N = (A>B)+(B>A).$$

We are interested to know with a probability of 95% the likelihood of succession, by comparing the magnitude of (A>B) versus (B>A). For example, if (A>B)=30, (B>A)=1 and (A>C)=60, (C>A)=2, which is the most likely successor of A: B or C? Although both ratios (A>B)/(B>A) and (A>C)/(C>A) equal 30, C is more likely than B to be the successor of A. Our measure gives in case of A and B a value of LM=0.85 and in case of A and C a value of LM=0.90, which is in line with our intuition.

Let us now consider again the examples from Figure 1. If we suppose that the number of lines in the log #L=1000, we can have three situations: (i) (C>E)=1000, (E>C)=0, LM= 0.997; (ii) (H>G)=600, (G>H)=400, LM=0.569; (iii) (F>B)=0, (B>F)=0, LM=0. In the sequential case (i), because always E succeeds C, LM≈1. When H and G are in parallel, in case (ii), LM=0.569, thus a value much smaller than 1. In the case of choice between F and B, in case (iii), LM=0. In general, we can conclude that LM can have a value (i) close to 1 when there is a clear tendency of succession between X and Y, (ii) in the neighborhood of 0.5 when there is both a succession between X and Y and between Y and X, but a clear tendency cannot be identified and (iii) zero when there is no succession relation between X and Y.

**The global metric GM.** The previous measure LM was expressing the tendency of succession by comparing the magnitude of (A>B) versus (B>A) at a *local level*. Let us consider that the number of traces in our log #L=1000, the frequency of events #A=1000, #B=1000 and #C=1000. We also know that (A>B)=900, (B>A)=0 and (A>C)=50 and (C>A)=0. The question is who is the most likely successor of A: B or C? For B, LM=0.996 and for C, LM=0.942, so we can conclude that they can be both considered as successors. However, one can argue that C succeeds A not as frequently, thus B should be considered a more likely successor. Therefore, we build the GM measure presented below.

$$GM = ((A>B)-(B>A))\frac{\#L}{\#A*\#B}.$$

Applying the formula above, we obtain for B as direct successor a value of GM=0.90, while for C, GM=0.05, thus B is more likely to directly succeeds A. In conclusion, for determining the likelihood of succession between two events A and B, the GM metric is indeed a global metric because it takes into account the overall frequency of events A and B, while the LM metric is a local metric because it compares the magnitude of (B>A) with (A>B).

**The causality metric CM.** The causality metric CM was first introduced in [11]. If for a given workflow log when task A occurs, shortly later task B also occurs, it is possible that task A causes the occurrence of task B. The CM metric is computed as following: if task B occurs after task A and $n$ is the number of events between A and B, then CM is incremented with a factor $(\delta)^n$, where $\delta$ is a causality factor, $\delta \in [0.0,1.0]$. We set $\delta$=0.8. The contribution to CM is maximal 1, if task B appears right after task A and then $n$=0. Conversely, if task A occurs after task B, CM is decreased with $(\delta)^n$. After processing the whole log, CM is divided by the minimum between the overall frequency of A and the overall frequency of B.

### 2.3 The dependency/frequency table

The starting point of our method is the construction of a so-called dependency/frequency (D/F) table from the workflow log information. An excerpt from the D/F table for the Petri net presented in Figure 1 is shown in Table 2. The information contained in the D/F table are: (i) the identifier for task A and B, (ii) the overall frequency of task A (#A), (iii) the overall frequency of task B (#B), (iv) the frequency of task B directly succeeded by another task A (B>A), (v) the frequency of task A directly succeeded by another task B (A>B), (vi) the frequency of B directly or indirectly succeeded by another task A, but before the next appearance of B (B>>>A), (vii) the frequency of A directly or indirectly succeeded by another task B, but before the next appearance of A (A>>>B), (viii) the local metric LM, (ix) the global metric GM and (x) the causality metric CM. The last column (DS) from Table 2 is discussed in the next section.

**Table 2**. Example of D/F table with direct succession (DS column) information. "T" means that task B is a direct successor of task a, while "F" means that B is not a direct successor of A

| AB | #A | #B | (B>A) | (A>B) | (B>>>A) | (A>>>B) | LM | GM | CM | DS |
|---|---|---|---|---|---|---|---|---|---|---|
| b a | 536 | 1000 | 536 | 0 | 536 | 0 | 0.00 | -1.0 | -1.0 | F |
| b b | 536 | 536 | 0 | 0 | 0 | 0 | 0.00 | 0.00 | 0.00 | F |
| b d | 536 | 279 | 0 | 279 | 0 | 279 | 0.99 | 1.86 | 1.00 | T |
| b j | 536 | 536 | 0 | 0 | 0 | 536 | 0.00 | 0.00 | 0.72 | F |
| b l | 536 | 1000 | 0 | 0 | 0 | 536 | 0.00 | 0.00 | 0.57 | F |
| b c | 536 | 257 | 0 | 257 | 0 | 257 | 0.99 | 1.86 | 1.00 | T |
| b e | 536 | 257 | 0 | 0 | 0 | 257 | 0.00 | 0.00 | 0.80 | F |

## 3 Data generation

For developing a model that will be used to decide when two events are in direct succession relation, we need to generate training data that resemble real workflow logs. Our data generation procedure consists on combinations of the following four possible elements that can vary from workflow to workflow and subsequently affect the workflow log:

- *number of event types*: we generate Petri nets with 12, 22, 32 and 42 event types.
- *amount of information* in the workflow log: the amount of information is expressed by varying the number of traces (one trace or line actually represents the processing of one case) starting with 1000, 2000, 3000, etc. and end with 10000 traces.
- *amount of noise*: we generate noise performing four different operations, (a) delete the head of a event sequence, (b) delete the tail of a sequence, (c) delete a part of the body and (d) interchange two random chosen events. During the noise generation process, minimal one event and maximal one third of the sequence is deleted. We generate three levels of noise: 0% noise (the common workflow log), 5% noise and 10% (we select 5% and respectively 10% of the original event sequences and we apply one of the four above described noise generation operations).
- *unbalance in AND/OR splits*: in Figure 1, after executing the event A, which is an OR-split, it is possible to exist an unbalance between executing tasks B and F. For example, 80%

of cases will execute task B and only 20% will execute task F. We progressively produced unbalance at different levels.

For each log that resulted from all possible combinations of the four elements mentioned before we produce a D/F table. In the D/F table a new field is added (the DS column) which records if there is a direct succession relationship between events A and B or not (True/False). An example of the D/F table with direct succession information is shown in Table 2. All D/F tables are concatenated into one unique and large final D/F/DS table that will be used to build the logistic regression model.

## 4 The logistic regression model

We have to develop a model that can be used to determine when two events A and B are in a direct succession relationship. The idea is to combine the three metrics described earlier and to find a threshold value $\sigma$ over which two events A and B can be considered to be in the direct succession relationship. In this section we develop a logistic regression model and we perform some validation tests.

The logistic regression estimates the probability of a certain dichotomic characteristic to occur. We want to predict whether "events A and B are in a direct succession relationship", that can be True/False. Therefore, we set as dependent variable the DS field from the D/F/DS file. The independent variables are the three metrics that we built earlier, i.e. the global metric GM, the local metric LM and the causality metric CM. In conclusion, we want to obtain a model that, given a certain combination of LM, GM and CM values for two events A and B, to predict the probability $\pi$ of A and B being in the direct succession relationship.

The form of the logistic regression is $log( \pi/(1-\pi) ) = B_0 + B_1*LM + B_2*GM + B_3*CM$, where the ratio $\pi/(1-\pi)$ represents the *odds*. For example, if the proportion of direct successors is 0.2, the odds equal 0.25 (0.2/0.8=0.25). The significance of individual logistic regression coefficients $B_i$ is given by the Wald statistics which indicates significance in our model; that means that all independent variables have a significant effect on direct succession predictability (Wald tests the null hypothesis that a particular coefficient $B_i$ is zero). The model goodness of fit test is a Chi-square function that tests the null hypothesis that none of the independents are linearly related to the log odds of the dependent. It has a value of 108262.186, at probability p<.000, inferring that at least one of the population coefficients differs from zero. The coefficients of the logistic regression model are shown in Table 3.

**Table 3.** Logistic analysis summary of three succession predictors of direct succession relation. The discrete dependent variable DS measures the question "are events A and B in a direct succession relationship?"; ** means significant at p<0.01

| Variables in the Equation* | B | Wald | df | Sig** | Exp(B) |
|---|---|---|---|---|---|
| LM | 6.376 | 2422.070 | 1 | .000 | 587.389 |
| GM | 4.324 | 920.638 | 1 | .000 | 75.507 |
| CM | 8.654 | 4490.230 | 1 | .000 | 5735.643 |
| Constant | -8.280 | 4561.956 | 1 | .000 | .000 |

Using the $B_i$ coefficients from Table 5, we can write the following expression *LR* from Eq. 1:

$$LR = -8.280 + 6.376*LM + 4.324*GM + 8.654*CM \quad (1)$$

Then the estimated probability $\hat{\pi}$ can be calculated with the following formula (Eq.2):

$$\hat{\pi} = e^{LR} / (1 + e^{LR}). \quad (2)$$

The influence of LM, GM and CM can be detected by looking at column Exp(B) in Table 3. For example, when CM increases one unit, the odds that the dependent =1 increase by a factor of ~5736, when the others variables are controlled. Comparing between GM, LM and CM, we can notice that CM is the most important variable in the model.

Inspecting the correct and incorrect estimates we can assess the model performance. Our model predicts the T value of DS in 95,1% of cases and the F value of DS in 99,2% cases. These values for correct/incorrect estimates are obtained at a cut value of 0.8, i.e. are counted as correct estimates those values that exceed 0.8. We set the cut value at 0.8, because we are interested in knowing the classification score when the estimated probability is high. Because 95% of the events which are in direct succession relationship are correctly predicted by the model, we conclude that we can set the threshold $\sigma = 0.8$. That means that we will accept that there is a direct successor relationship between events A and B, if the estimated probability would exceed 0.8. The following step is to test the model performance on test material.

**Model testing**. We describe two different type of tests: (i) k-fold cross-validation on test material extracted from the learning material and (ii) model check on a completely new test material.

*K-fold cross-validation* (k-fold cv) is a model evaluation method that can be used to see how well a model will generalizes to new data of the same type as the training data. The data set is divided into *k* subsets. Each time, one of the *k* subsets is used as the test set and the other *k-1* subsets are put together to form a training set. Then the average error across all *k* trials is computed. Every data point gets to be in a test set exactly once, and gets to be in a training set *k-1* times. The variance of the resulting estimate is reduced as *k* is increased. We take k=10. The results of our 10-fold cv gives for the 10 training sets an average performance of 95.1 and for the 10 testing sets an average performance of 94.9, so we can conclude that our model will perform good in case of new data.

In order to test the model performance on completely new data, we build a new more complex Petri net with 33 event types. This new PN has 6 OR-splits, 3 AND-splits and three loops (our training material contains Petri nets with at most one loop). We consider three Petri nets with three different levels of unbalance and using the formula from Eq. 2, we predict the probability of direct succession for the Petri net. For these three Petri nets, we counted the number of direct successors correctly found with our method. The average of direct successors that were correctly found is 94.3. Therefore we can conclude that even in case of completely new data, i.e. a workflow log generated by a more complex Petri net, the method has a good performance of determining the direct successors.

## 5 Conclusions and future directions

Using the presented method, we developed a model that estimates the probability that two events A and B are in the direct successor relation. The model performance is good, i.e. 95% of the original direct succession relations were found. However, it is interesting to investigate what is the reason that the rest of 5% direct connections were not discovered. Inspecting

these cases, we notice that although between event A and B there is a direct succession relation, the value of (A>B) is too small, and subsequently, the values for all three metrics are also small. To illustrate such a situation, inspect Figure 1. If we suppose that event H is always processed in 1 time unit, event G in 3 time units and I in 2 time units and H always finishes its execution before I starts, then we will always see the sequence "AFHGIKL" and never the sequence "AFGIHKL". Although K is the direct successor of H, the method will not find the connection between H and K.

In conclusion, we presented a global learning method that uses information contained in workflow logs to discover the direct successor relations between events. The method is able to find almost all direct connections in the presence of parallelism, noise and an incomplete log. Also, we tested our model on a workflow log generated by a more complex Petri net than the learning material, resulting in a performance close to that of the first experiment.

We plan to do future research in several directions. First, because in many applications, the workflow log contains a timestamp for each event, we want to use this additional information to improve our model. Second, we want to provide a method to determine the relations between the direct successors and finally to construct the Petri net.

## References


[1] W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. J. of Circuits, Systems, and Computers, 8(1): 21-66, 1998.

[2] R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process models from Workflow Logs. In Sixth International Conference on Extended Database Technology, pg. 469-483, 1998.

[3] J.E. Cook and A.L. Wolf. Discovering Models of Software Processes from Event-Based Data, ACM Transactions on Software Engineering and Methodology, 7(3):215-249, 1998.

[4] J.E. Cook and A.L. Wolf. Event-Based Detection of Concurrency. In Proceedings of the Sixth International Symposium on the Foundations of Software Engineering (FSE-6), Orlando, FL, November 1998, pp. 35-45.

[5] J.E. Cook and A.L. Wolf. Software Process Validation: Quantitatively Measuring the correspondence of a Process to a Model. ACM Transactions on Software Engineering and Methodology, 8(2): 147-176, 1999.

[6] J. Herbst. A Machine Learning Approach to Workflow Management. In 11th European Conference on Machine Learning, volume 1810 of Lecture Notes in Computer Science, pages 183-194, Springer, Berlin, Germany, 2000.

[7] J. Herbst. Dealing with Concurrency in Workflow Induction In U. Baake, R. Zobel and M. Al-Akaidi, European Concurrent Engineering Conf., SCS Europe, Gent, Belgium, 2000.

[8] J. Herbst and D. Karagiannis. An Inductive approach to the Acquisition and Adaptation of Workflow Models. In M. Ibrahim and B. Drabble, editors, Proceedings of the IJCAI'99 Workshop on Intelligent Workflow and Process Management: The New Frontier for AI in Business, pg. 52-57, Stockholm, Sweden, August 1999.

[9] J. Herbst and D. Karagiannis. Integrating Machine Learning and Workflow Management to Support Acquisition and adaptation of workflow Models. International Journal of Intelligent Systems in Accounting, Finance and Management, 9:67-92, 2000.

[10] L. Maruster, W.M.P. van der Aalst, T. Weijters, A. van den Bosch, W. Daelemans. Automated discovery of workflow models from hospital data. In Kröse, B. et al. (eds.): Proceedings 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC'01), 25-26 October 2001, Amsterdam, The Netherlands, pp. 183-190.

[11] T. Weijters, W.M.P. van der Aalst. Process Mining: Discovering Workflow Models from Event-Based Data. In Kröse, B. et. al, (eds.): Proceedings 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC'01), 25-26 October 2001, Amsterdam, The Netherlands, pp. 283-290.