An Organizational Mining Approach Based on Behavioral Process Patterns

Completed Research Paper

Jie Tao
Dakota State University
Madison, South Dakota, USA
jtao16065@pluto.dsu.edu

Amit V. Deokar
Pennsylvania State University
Erie, Pennsylvania, USA
amit.deokar@psu.edu

Introduction

Process aware information systems (PAIS) and components have become ubiquitous in different business domains and areas of business computing (Dumas et al. 2005). They vary from being full-fledged enterprise systems (e.g., business process management systems, enterprise resource planning) that coordinate entire business processes to standalone applications or systems that capture execution information related to certain process segments or tasks. PAIS embed key process knowledge nuggets in the form of process-related observations, also termed as *event logs* that can be mined using analytical methods. Process Mining and Analytics (PMA) facilitates discovery, monitoring, and improvement of current existing processes based on the analysis on event logs (van der Aalst 2012). PMA techniques can be applied to various phases of a business process lifecycle to facilitate organizational learning based on process-related knowledge. PMA have been recognized to augment current Business Intelligence (BI) techniques by investigating activities inside the business processes rather than treating them as '*black boxes*', and thus addressing a limitation of mainstream BI applications of mainly focusing on aggregating data for assisting high-level tactical or strategic decision making (Bucher et al. 2009).

While researchers have primarily focused on applying PMA techniques for studying control-flow perspective of business processes, there is limited work on their organizational perspective (van der Aalst 2011). Organizational and social structures underlying business processes need to be understood well for improving organizational processes. Existing research relies on explicit process models and/or organizational models for obtaining information on organizational settings and interactions among knowledge workers (Song and van der Aalst 2008; Thomas and Fellmann 2006). However, in many cases such as inter-organizational workflows and cross-department team-based processes, neither are process/organizational models explicitly available, nor does a formalized structure exist to represent such knowledge. Thus, an alternative approach is deemed necessary.

This study focuses on organizational mining, which entails extracting information on organizational structures/models and essentially providing insights on how knowledge workers interact to perform business processes. In particular, this work contributes to the PMA body of knowledge by proposing: a) a pattern recognition and matching approach in the form of *Behavioral Pattern Discovery Algorithm (BPDA)* for identifying behavioral process patterns from event logs, b) a novel organizational mining approach consisting of *Org-AHC* algorithm based on behavioral process patterns, and c) the *OrgMiner* framework that encapsulates three modules, namely *pattern definition*, *pattern selection*, and *organizational mining* in a systematic manner through links to various pertinent knowledge elements. These design artifacts have been validated through a case study consisting of a large event log, which also demonstrate their feasibility and utility.

The structure of this paper is organized as follows. Section 2 provides background and overview of PMA, with particular focus on the organizational perspective. A motivating and running example used throughout the paper is discussed in Section 3. In Section 4, key design artifacts including the *OrgMiner* framework, behavioral pattern recognition using the *BPDA* algorithm, and organizational mining approach using the *Org-AHC* algorithm are presented. In section 5, the design artifacts are validated through a case study. Discussion on related work, situating the key contributions of this study, is presented in Section 6. The article concludes with summary remarks in Section 7.

Background

Process Mining and Analytics

Process mining and analytics techniques can be used in conjunction with any PAIS, as long as the observations of the actual executions are recorded as *event logs* – which can be segmented into *events* (the atomic steps in a specific process) or *instances* (the *ad-hoc* executions of a specific process). Also, Pérez-Castillo et al. (2011) have recently proposed an approach obtaining event logs from non-PAIS (without logging functionalities). The essence of PMA activities is to mine event logs to extract and present embedded knowledge nuggets in a user-understandable form. PMA has been recognized as *evidence-based* analysis since all the analyses are based on *facts* (van der Aalst 2012). Given that PMA has its roots in the data mining field, its practices are analogous to data mining applications. Notably, data mining techniques derive data patterns (as formal abstractions of events in the real world) from data sets (as factual records of observations regarding the events), while PMA tools mine process execution data (event logs, as observations of events in certain business processes) to construct the process models (as abstractions of actual business processes) (Tiwari et al. 2008). The key similarity between data mining and PMA is that they both derive *explicit* knowledge hidden in (large) data collections.

Earlier PMA approaches have been based on heuristic algorithms, which identify business process patterns based on "*rule-of-thumb*" assumptions (e.g. the *α-algorithm* (van der Aalst et al. 2004)). Although, a growing number of PMA methods make use of advanced analytic techniques such as genetic algorithms (van der Aalst et al. 2005), fuzzy logic algorithms (Günther and van der Aalst 2007), temporal analysis based algorithms (Köck and Paramythis 2011; Veiga and Ferreira 2010), clustering algorithms (Bose and van der Aalst 2010; Song et al. 2009), and pattern recognition based algorithms (Bae et al. 2006; Ferreira and Thom 2012; Smirnov and Weidlich 2009).

PMA activities can be broad characterized along two dimensions, namely *functions* and *perspectives*. In regards to the functional dimension, PMA techniques entail three major functions: *discovery*, *conformance checking*, and *enhancement*. The *discovery* function entails mining the event logs to (semi-) automatically constructing underlying process models and associated properties (organizational properties and data objects), and has been focus of traditional PMA approaches (van Dongen and van der Aalst 2004). The *conformance checking* function involves deviation detection by comparing the mined process model (from executed process observations) with an a priori model (Bose and van der Aalst 2012). The *enhancement* function also involves an a priori model; but rather than checking the conformance, it extends the mined process model with a new aspect or perspective (Jareevongpiboon and Janecek 2013). Popular enhancement activities include decision mining, user profiling, and performance analysis. The perspective dimension of PMA activities pertain to inquiring about various aspects of business processes such as the *process* perspective ('How'), the *resource* perspective ('Who'), and the *data* perspective ('What') (Weske 2012). The process perspective focuses on the control-flow, i.e., ordering of events in business processes (e.g. (Bertolini et al. 2011)); the resources perspective focuses on the originators of the events and how they are related (e.g. (Song and van der Aalst 2008)); the data perspective focuses on the data objects and their values associated with each of the executed events (e.g. (Sun and Zhao 2013)).

Majority of extant PMA studies focus on the *process* perspective, with all three functions, while the classical data mining and BI (e.g. OLAP) area concentrates on the *data* perspective when applied to the BPM context. Also, prior studies regarding the *resource* perspective of the business processes rely highly on explicit knowledge abstractions such as process and/or organizational models (van der Aalst and Song 2004; Sellami et al. 2013). Notably, there is lack of research on techniques for mining organizational information solely from event logs, in absence of other explicit process or organizational structure representations. In this research study, we contribute towards addressing this research gap.

Organizational Mining

Organizational mining refers to the PMA activities aimed toward analyzing implicit information about resources in event logs (van der Aalst and Song 2004). Organizational mining has two key goals: (1) to develop an understanding of the social networks depicting the interactions among different resources/originators in (cross-)organizational processes, and (2) to (re)structure cross-functional and/or cross-organizational teams of knowledge workers based on their organizational roles or units (IEEE Task

Force on Process Mining 2011). Analogous to PMA, organizational mining also entails functions related to discovery, *conformance checking*, and *enhancement*. *Discovery* in organizational mining refers to extraction of organizational models or social network models that reflect the organizational settings from the information within event logs. Organizational models usually contain information on organizational units (e.g., departments), roles (e.g., jobs), originators (e.g., employees), and relationships (e.g., A is "*part-of*" B, or C "*is-a*" D), while social network models depict the interactions (e.g., *handover of work*, etc.) among different originators. Also, organizational mining can be used in tandem with other related information to discover job assignment rules, resource allocation rules, or user profiling rules. *Conformance checking* can be performed within the organizational mining context by comparing discovered models/rules with corresponding a priori models/rules. Further, *enhancement* in the context of organizational mining refers to enriching existing model/rules with additional information, such as providing abstracted process models based on different organizational units/roles. Organizational mining provides a new dimension for PMA, which can provide the management team with analytical insights on the organizational context. This includes data-driven information regarding the organizational structure and communication as well as better understanding of business process performance indicators from an organizational operations management perspective. Such information is difficult to be extracted through traditional PMA activities.

Researchers in the PMA domain have recognized the importance of organizational modeling and have proposed various approaches. Sellami et al. (2013) have proposed the use of an organizational ontology for semantically annotating event logs to discover relationships among task performers. This approach has the limitation of relying on a static underlying knowledge structure, i.e. the organizational ontology. Song and van der Aalst (2008) propose an approach that is primarily aimed at conformance checking of organizational models, and as such relies on existing explicit process and organizational models. Their approach relies on frequencies of different originators executing similar actions to recommend grouping of such originators in the same organizational unit. In this study, we take a discovery-oriented approach to organizational mining, where we do not assume that the existence of explicit models (either process models or organizational models) given that such models may be inaccessible in practice (e.g., in cross-organizational processes). Further, in our proposed approach we not only use task execution frequencies of different originators, but also rely on recurring behavioral patterns of process segments within the event logs to draw inferences regarding associations among originators and ultimately infer an underlying organizational model. In using process patterns within our approach, we build on the extant work in the workflow patterns area including Smirnov et al.'s (2012) action patterns notions used for process mining.

A Running Example

To illustrate the concepts described in the proposed framework in a self-contained manner, we use a portion of the event log from the "reviewing process" documented in van der Aalst (2011). The example describes a reviewing process of an academic journal. When a paper is submitted to the journal, it is sent to three different reviewers ('*invite reviewers*', Activity A). Each of the reviewers is responsible to write some comments about the paper ('*get review 1-3*', Activity B1-3). A person is going to read the reviewers' comments and make decision ('*decide*', Activity D) after the comments are collected ('*collect reviews*', Activity C). However, sometimes reviewers do not respond to the reviewing requests. In those cases, additional reviewers are invited ('*invite additional reviewer*', Activity E). The additional reviewer comments on the paper ('*get review X*', Activity F) and then the comments are read to make decision ('*decision*', Activity D). These steps (E-F-D) are repeated until enough comments are collected. Then a final decision is made ('*accept*', Activity G or '*reject*', Activity H).

The original event log contains 100 instances (papers) and 3,730 events (in 8 distinct event classes). For the example, we randomly selected 6 instances shown in Table 1 (each row corresponds to a different instance, which is identified by the *instance ID*), which contain all 8 event classes. We denote each event by two elements: the activity (e.g. '*C*') and its originator (e.g. '*Anne*'). In total 10 originators are involved in these 6 instances. The events in each instance are ordered according to the temporal sequence reflected in their timestamps. The selected event log is pre-processed in two steps. First, since the *transition changes* of the events are less relevant to the organizational mining purpose, we decide to ignore them and choose all '*completed*' events. Next, we filter out the events without originator information (i.e. the originators of all the '*time-out*' events are tagged as *'_INVALID_'*, thus all such events are excluded from the example).

Besides the selected event log (from now noted as "*example log*") shown in Table 1, we assume that the underlying process model and the organizational model are unknown/implicit. For illustration purpose, we assume the 6 instances are exhaustive of the process (no other instance exists). In the remaining sections, we use this example to illustrate aspects of the proposed organizational mining approach.

Instance ID	Process Activities and Performers
1	(A, Mike), (B2, Carol), (B3, Pam), (B1, John), (C, Anne), (D, Wil), (E, Anne), (F, Pam), (D, Wil), (E, Mike), (F, Mary), (D, Wil), (E, Mike), (F, Sara), (D, Wil), (H, Anne)
10	(A, Anne), (B2, Sara), (C, Mike), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (D, Wil), (E, Anne), (F, Pete), (D, Wil), (E, Anne), (D, Wil), (E, Mike), (F, Carol), (D, Wil), (E, Anne), (D, Wil), (G, Mike)
11	(A, Mike), (B3, Sam), (C, Anne), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (D, Wil), (E, Anne), (F, Mary), (D, Wil), (E, Anne), (D, Wil), (E, Anne), (F, Mary), (D, Wil), (E, Mike), (F, Carol), (D, Wil), (H, Anne)
12	(A, Anne), (B2, Carol), (B3, John), (C, Anne), (D, Wil), (E, Mike), (D, Wil), (E, Anne), (D, Wil), (E, Mike), (F, Pete), (D, Wil), (E, Anne), (F, Pete), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (F, Carol), (D, Wil), (H, Anne)
13	(A, Anne), (B2, Carol), (B3, John), (C, Anne), (D, Wil), (E, Anne), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (F, Carol), (D, Wil), (E, Anne), (D, Wil), (E, Mike), (D, Wil), (E, Mike), (F, Pam), (D, Wil), (E, Anne), (F, Pete), (D, Wil), (E, Mike), (F, Carol), (D, Wil), (H, Mike)
100	(A, Anne), (B3, Pam), (B1, Sara), (B2, Pete), (C, Anne), (D, Wil), (E, Mike), (D, Wil), (E, Anne), (D, Wil), (E, Anne), (F, Mary), (D, Wil), (E, Mike), (D, Wil), (E, Anne), (F, Mary), (D, Wil), (G, Mike)

Table 1. Event Log for the Running Example

Design and Development of the *OrgMiner* Framework

We first formally define the foundational notion of event logs.

Definition 1: (Event Logs). Let $\mathcal{A}$ ($\mathcal{A} = \{a_1, \dots, a_n\}, n = |\mathcal{A}|$) be the set of activities in a business process, and $\mathcal{O}$ ($\mathcal{O} = \{o_1, \dots, o_m\}, m = |\mathcal{O}|$) be the set of all distinct originators (e.g. persons, system modules) that conduct the activities in $\mathcal{A}$. $\mathcal{E} = \mathcal{A} \times \mathcal{O}$ is the cartesian product that represents a set of events indicating association of activities and originators (e.g. (D, Wil) implies that activity D is conducted by originator Wil). $\mathcal{E}^+$ is the set of possible non-empty finite ordered sequence of events from $\mathcal{E}$. Any $t \in \mathcal{E}^+$ is a possible trace (temporally ordered sequence of events). An *event log* $\mathcal{L}$ is a multi-set (bag) of traces. Each $l \in \mathcal{L}$ is a process instance (an one-time execution of the process). We define the following related notions:

- $\mathcal{R} = [r_{i,j}]$ is a symmetric, non-empty matrix that reflects the relations between each pair of activities in $\mathcal{A}$; i.e. $r_{i,j} = (a_i, a_j), \forall\, a_i, a_j \in \mathcal{A}$;
- π_A is an assignment operation on originators that returns all the activities executed by the same originator, e.g. $\pi_A(o_i) = \{a_i | a_i \in \mathcal{A}, i = p, \dots, q\}$;
- Let $e = (a, o) \in \mathcal{E}$ be an event, π_E is an operation that retrieves the activity from the given event, e.g. $\pi_E(e) = a \in \mathcal{A}$.

In defining the notion of behavioral process patterns later, we will make the following assumptions on the event logs: (1) Each of the events in an event log has an originator; events without any originator are excluded. (2) Each of the events signifies an action; events not referring to an action are discarded as erroneous from the event log(s). (3) If the activities are highly related, then the relatedness among originators executing them is also high.

OrgMiner Framework

The proposed *OrgMiner* framework is illustrated in Figure 1. The proposed framework is aimed at business process analysts as end users and is intended to provide decision support when conducting organizational analysis (e.g. *learning implicit organizational knowledge from process-aware systems*). Figure 1 depicts the information system modules of the framework, as well as the knowledge elements used and generated by these modules. The *OrgMiner* framework consists of three modules, namely: *Pattern Definition*, *Pattern Selection*, and *Organizational Mining*. Additionally, a dashboard acts as an interface between the system and the end users.

Within the *Pattern Definition* module, three basic types of behavioral patterns are articulated: strict order pattern, exclusiveness pattern, and interleaving pattern. They are represented as 'pattern schemas'. Users can create more complex or ad hoc patterns based on them. The event log(s) to be analyzed serves as an input to this module. The *Pattern Selection* module relies on the pattern schemas and the event log(s) to generate the 'behavioral relation matrix' using the proposed *Behavioral Pattern Discovery Algorithm (BPDA)*. This matrix essentially captures the instantiation of patterns in the event logs. Selection of patterns is based on support and confidence as metrics, resulting in (selected) 'behavioral patterns'. The *Organizational Mining* module utilizes the discovered behavioral patterns to create 'organizational analytical reports'. In this module, activity distances, activity relatedness, and originator relatedness are used as metrics to align the relatedness between activities to the relationships between originators. Following this, the module utilizes the proposed *Organizational Mining Algorithm (Org-AHC)* for discovering the organizational model, essentially by clustering originators with higher originator relatedness values together in a stepwise manner. Based on the discovered organizational model, organizational analytical reports can be generated through (semi-)structured queries for *discovery*, *conformance checking*, and/or *enhancement* purposes. Each of the three modules of the OrgMiner framework and their interaction with the knowledge elements is discussed in detail next.

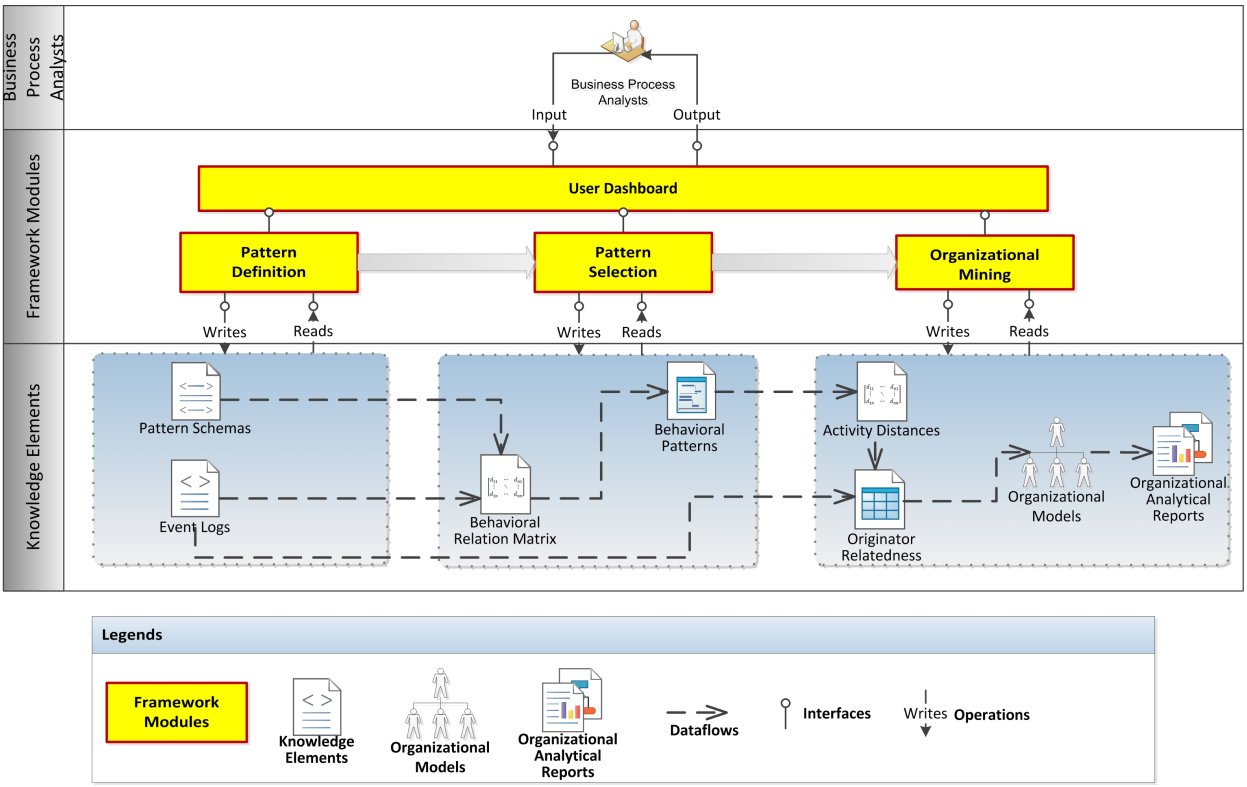


Figure 1. *OrgMiner* Framework

Module I: Pattern Definition

The *Pattern Definition* module concerns tasks such as defining basic behavioral patterns, as well as defining *ad hoc*, more complex patterns based on the basic behavioral patterns. In this article, we restrict our scope to basic behavioral patterns. The general notion of patterns in the BPM context has been elaborated by van der Aalst and ter Hofstede (2003). Analogous to action patterns proposed by Smirnov et al. (2012), behavioral patterns rely on the concept of behavioral relations defined in terms of weak order relations (Weidlich et al. 2011). The major difference is that the action patterns are defined for business process model repositories, while our behavioral patterns are defined with respect to the event logs (with no explicit process model).

Definition 2: (Weak Order Relations). Let $\mathcal{L}$ be an event log, and $\mathcal{E}^+$ be the set of all possible traces in $\mathcal{L}$. The *weak order relation* ($\succ_{\mathcal{L}} (a_i, a_j)$) contains all pairs of (a_i, a_j) such that $\exists\, t \in \mathcal{E}^+$, if and only if $\pi_E(e_i) = a_i \wedge \pi_E(e_j) = a_j$, $1 \le i < j \le |t|$, where t is an ordered trace in $\mathcal{L}$.

We denote a *weak order relation* as $r_{i,j} = \succ_L^d (a_i, a_j)$, $a_i, a_j \in \mathcal{A}$. The integer d denotes the distance between the activities a_i and a_j in a process instance l_k ($l_k \in \mathcal{L}$) and is measured by the transitions between the two activities. Notably, the *weak order relation* is not transitive, i.e. $\succ_L^d (a_i, a_j) \neq \succ_L^d (a_j, a_i)$. Further, $r_{i,j} = \not\succ_{\mathcal{L}} (a_i, a_j)$ denotes that a *weak order relation* does not exist between a_i and a_j.

We now define three types of behavioral relations, the type of which depends on how the weak order relation exists in a pair of activities in an event log.

Definition 3: (Behavioral Relations). Let $\mathcal{L}$ be an event log, a *Behavioral Relation* $r_{i,j} \in \mathcal{R}$ has to be one of the following:

- *Strict order relation* ($\rightarrow_L^d (a_i, a_j)$), if and only if $a_i \succ_{\mathcal{L}} a_j \wedge a_j \not\succ_{\mathcal{L}} a_i$;
- *Exclusiveness relation* ($+_L^d (a_i, a_j)$), if and only if $a_i \not\succ_{\mathcal{L}} a_j \wedge a_j \not\succ_{\mathcal{L}} a_i$;
- *Interleaving relation* ($\|_L^d (a_i, a_j)$), if and only if $a_i \succ_{\mathcal{L}} a_j \wedge a_j \succ_{\mathcal{L}} a_i$.

The strict order relation is not transitive, while the other two behavioral relations are transitive. These *behavioral relations* are codified as *pattern schemas*, and used in subsequent modules.

The use of such behavioral relations can be illustrated with the activities in the example logs. For instance, it exists $\rightarrow_L^d (\mathrm{A}, \mathrm{D})$ because in all instances Activity A precedes Activity D. Similarly, it also holds $\|_L^d (\mathrm{D}, \mathrm{E})$ and $+_L^d (\mathrm{G}, \mathrm{H})$. Notably, the strict order relation is not transitive, while the other two behavioral relations are. In instance 1, it exists $\rightarrow_L^3 (A, C)$ since there are three activities between them. If an activity appears several times in a process instance (e.g. in a loop), d is the longest distance between the two activities. For instance, in instance 10, it exists $\rightarrow_L^{16} (A, D)$ since maximally there are 16 activities between A and D.

Module II: Pattern Selection

The second module, *Pattern Selection*, utilizes the event log(s) and the pattern schemas from the prior module and identifies the behavioral patterns from the event log(s). It generates the behavioral relation matrix as an intermediate output for further application purposes.

Before we proceed to the pattern discovery algorithm, we have to define *the Behavioral Relation Matrix* (Definition 4).

Definition 4: (Behavioral Relation Matrix). Let $\mathcal{L}$ be an event log. $R^B = \mathcal{A} \times \mathcal{A}, R^B \subseteq \mathcal{R}$ is a non-empty matrix that reflects the behavioral relations and their occurrences between each pair of activities in $\mathcal{A}$; e.g. $r_{i,j}^B = (r_{i,j}, n), \forall\, r_{i,j}^B \in R^B$, where $r_{i,j}$ is the *behavioral relation* between activities (a_i, a_j) as defined above, while n is the number of occurrences, i.e. frequency of *behavioral relation* $r_{i,j}$ in $\mathcal{L}$.

The Behavioral Pattern Discovery Algorithm (BPDA) illustrated in Figure 2(a) is used to generate the Behavioral Relation Matrix for all the pair of activities from an event log, across all its instances. As an illustration, a portion of the Behavioral Relation Matrix consisting of activities C, D, E, and F with respect to the strict order relation is shown in Figure 2(b).

In order to determine which of the discovered patterns should be selected, we use two metrics, namely support and confidence as defined next in Definition 5. For deriving the pattern selection threshold, we given put more emphasis on *confidence* as since it indicates which relation is of more significance in relation to others.

Definition 5: (Support and Confidence). Let us denote the support of a particular behavioral pattern $supp(r_{i,j})$ as the frequency of such pattern appearing between activities (a_i, a_j) across all process instances ($l_k \in \mathcal{L}$) in an event log $\mathcal{L}$ (which is n according to Definition 4 above). Confidence of the

behavioral pattern is defined as $conf(r_{i,j}) = \frac{supp(r_{i,j})}{k}$ $(k \neq 0)$, where k is the number of the process instances in the given event log.

Assuming the frequencies of $\rightarrow_L^d (a_i, a_j)$, $+_L^d(a_i, a_j)$, and $\|_L^d (a_i, a_j)$ as n_1, n_2, and n_3, respectively, it can be noted that $k = n_1 + n_2 + n_3$, since the three behavioral relations are mutually exclusive to each other and together comprise the complete event log. As a result, the confidence can indicate the strength of the respective relation. As an example, $supp(+_L(\mathrm{B1}, \mathrm{H})) = 5$, while $conf(+_L(\mathrm{B1}, \mathrm{H})) = \frac{5}{1+0+5} = 5/6$.

Algorithm 1: Behavioral Pattern Discovery Algorithm (BPDA)

INPUT: $l \in \mathcal{L}$ -- Process Instances.
OUTPUT: $\mathcal{R} = [r]$ – Behavioral Relation Matrix.

```
1  //Initialize:
2  R ← φ;
3  i, j = 0;
4  //Begin:
5  FOREACH (l ∈ L)
6    FOREACH (a_i, a_j ∈ l)
7      IF (a_i precedes a_j) THEN
8        r_{i,j} =→ (a_i, a_j);
9      END IF;
10     NEXT;
11     IF (a_j precedes a_i) || (r_{i,j} ¬ empty) THEN
12       r_{i,j} =|| (a_i, a_j);
13     END IF;
14   END FOR;
15   FOREACH (r_{i,j} ∈ R)
16     IF (r_{i,j} empty) THEN
17       r_{i,j} = +(a_i, a_j);
18     END IF;
19   END FOR;
20 END FOR;
21 RETURN R; //Return the aggregated behavioral relations across different instances
22 //End.
```

(a) The BPDA Algorithm

→	C	D	E	F
C	-			
D	(→, 6)	-		
E	(→, 6)	(→, 0)	-	
F	(→, 6)	(→, 0)	(→, 0)	-

(b) The Example Output from BPDA

Figure 2. (a) Behavioral Pattern Discovery Algorithm and (b) Example Output (partial)

The BPDA algorithm provides the support and confidence of the patterns between all pairs of activities in $\mathcal{L}$. Then, if the supports and confidences of the strict order/interleaving patterns are greater than the user-defined thresholds, the patterns are accepted; otherwise, the exclusiveness pattern holds.

There are several methods to determine the threshold(s) for pattern selection purpose, for instance: i) a user-defined value; ii) an (semi-)automatic procedure to determine the optimistic values for the threshold(s). In this article, we use a combination of both to determine the pattern selection thresholds, as discussed below. For the pattern selection purpose, we are interested in activity sets with high support and confidence values; thus, we require any selected pattern to have the highest *confidence* value (comparing to other behavioral patterns between the same pair of activities). The pattern selection process is similar in theme to the a priori algorithm for association rules proposed in (Agrawal and Srikant 1994; Savasere et al. 1995), but the definitions of the support and confidence metrics are unique to this application.

Based on the behavioral relations, we can now define the behavioral patterns as follows.

Definition 6: (Behavioral Patterns). We use a tuple $BP = (r_{i,j}, d, conf)$ to define a behavioral pattern in an event log $\mathcal{L}$, where:

- $r_{i,j}$ is one of the behavioral relations between (a_i, a_j), as defined in Definition 3;
- d is the distance between the two activities, as defined in Definition 3;
- $conf$ is the confidence of the specific pattern between (a_i, a_j), which is defined in Definition 5.

We use the example log to illustrate the pattern discovery process. Based on the above definitions and algorithm, the selected Behavioral Patterns are shown in Table 2.

In Table 2, the matrix is asymmetric by its diagonal elements. Due to the space limitation, we are not able to show the overall process of obtaining the Behavioral Relation Matrix. In essence, however, we run

algorithm 1 on the example log (for each instance), and select behavioral patterns with the highest confidence value.

		Predecessors									
	Task	A	B1	B2	B3	C	D	E	F	G	H
Successors	A	-	-	-	-	-	-	-	-	-	-
	B1	+	-	-	-	-	-	-	-	-	-
	B2	→	→(-1)	-	-	-	-	-	-	-	-
	B3	→	+	||	-	-	-	-	-	-	-
	C	→	+	→	→	-	-	-	-	-	-
	D	→	+	→	→	→	-	-	-	-	-
	E	→	+	→	→	→	||	-	-	-	-
	F	→	+	→	→	→	||	||	-	-	-
	G	+	+	+	+	+	+	+	+	-	-
	H	→	+	→/+	→	→	→	→	→	+	-

Note:
1) → refers to strict order patterns, || refers to interleaving patterns, + refers to exclusiveness patterns;
2) The patterns of an activity itself do not exist (e.g. $\nexists(A \Rightarrow A)$);
3) We only consider the behavioral relations along the temporal order (e.g. (A,B1), not vice versa).

Table 2. Example of Selected Behavioral Patterns

As shown in Table 2, two cellblocks are highlighted ($B1 \Rightarrow B2$ and $B2 \Rightarrow H$). The cell $B1 \Rightarrow B2$ is highlighted since there is only a *reversed* strict order pattern between (B1, B2): the behavioral relation $\rightarrow_L^d (B1, B2)$ does not stand while $\rightarrow_L^d (B2, B1)$ holds in the given instances in the running example, By definition, the strict order relation is not transitive – so that we use the symbol $\rightarrow (-1)$ to denote the reversed strict order relation. The cell $B2 \Rightarrow H$ is highlighted since both the values of supports and confidences of the strict order pattern and the exclusiveness pattern are equal. Thus, additional measures need to be undertaken. The business process analysts can either manually assign a behavioral pattern to the pair of activities, or create a new pattern style of interest, in order to resolve such conflicts. In this example, we define that a reversed strict order pattern is equivalent to a strict order pattern, and select the strict order pattern over the exclusiveness pattern between (B2, H) for the organizational mining purpose.

Module III: Organizational Mining

We use the behavioral patterns discussed in the section above as the basis of mining organizational structure from the event logs. From the pattern matching and selection module, we have the Behavioral Relation Matrix $R^B = [r^B]$, which is an $|A| \times |A|$ matrix. To quantify and normalize the relations between each pair of activities (a_i, a_j), we can use the *Activity Distance Matrix* defined as follows.

Based on the activity distance d introduced in Definition 4, we formalized the average behavioral distance in an event log $\mathcal{L}$ (across different process instances) as in Definition 7.

Definition 7: (Average Behavioral Distances). Let $T(a_i, a_j)$ be the set of all directional sequences containing activities a_i, a_j. $\forall t_k(a_i, a_j) \in T(a_i, a_j)$ (k = 1, 2, …, m), if the distance between a_i and a_j is d_k. Then the average behavioral distance ($\overline{d(a_i, a_j)}$) between a_i, a_j is given by: $\overline{d(a_i, a_j)} = \frac{\sum_1^m d_k}{m}$.

For instance, from the example log, $\overline{d(A, C)} = \frac{1+1+1+2+2+3}{6} = \frac{5}{3}$.

We can now compute the *activity distance* based on the *average behavioral distance* using the following logic. If an interleaving pattern holds between (a_m, a_n), the activity similarity between (a_m, a_n) would be the highest, so the activity distance between them would be the lowest (for computational purpose, denote as 0). If an exclusiveness pattern exists between (a_m, a_n), the activity similarity between (a_m, a_n) would be the lowest, so the activity distance between them would the highest (for computational purpose, denote as $+\infty$). If a strict order pattern holds between (a_m, a_n), then $\mathcal{D}(a_m, a_n) \in (0, +\infty)$. Since $\mathcal{D}(a_m, a_n)$ increases as $\overline{d(a_m, a_n)}$ increases and $conf(r_{m,n})$ decreases, we computationally denote $\mathcal{D}(a_m, a_n) = \frac{\overline{d(a_m, a_n)}}{conf(r_{m,n})}$. Based on this rationale, the activity distance is defined as follows (Definition 8).

Definition 8: (Activity Distance). The *Activity Distance* $\mathcal{D}$ is defined as follows:

$$\mathcal{D}(a_m, a_n) = \begin{cases} 0 \ \textit{if an interleaving pattern exists between } (a_m, a_n) \\ \left(\dfrac{\overline{d(a_m, a_n)}}{conf(r_{m,n})}\right) \textit{if a strict order pattern exists between } (a_m, a_n) \\ +\infty \textit{ if an exclusiveness pattern exists between } (a_m, a_n) \end{cases}$$

where, *conf* is the confidence of a behavioral pattern between a pair of activities (a_m, a_n); and $\bar{d}$ is the average distance between the two activities in the same pattern across different instances.

The definition of $\mathcal{D}$ is straightforward and it is represented in a matrix format. With it defined, we can define the *Activity Relatedness* as following (Definition 9). Since the activity distances are located by $[0, +\infty]$, we use the exponent function of e to normalize $AR(a_m, a_n) \in [0, 1]$. Moreover, we define AR of the same activity equals 1, e.g. $AR(a_m, a_m) = 1$.

Definition 9: (Activity Relatedness). We define the *Activity Relatedness* (*AR*) as:

$$AR(a_m, a_n) = e^{-\mathcal{D}(a_m, a_n)} = \begin{cases} e^0 = 1 \ \textit{if an interleaving pattern exists between } (a_m, a_n) \\ e^{-\frac{\overline{d(a_m, a_n)}}{conf(r_{m,n})}} \quad \textit{if a strict order pattern exists between activities} \\ e^{-\infty} = 0 \textit{ if an exclusiveness pattern exists between activities} \end{cases}$$

In Definition 9, e is the mathematical constant.

AD	A	B1	B2	B3	C	D	E	F	G	H
A										
B1	+∞									
B2	0.48	+∞								
B3	0.72	+∞	0							
C	2.00	+∞	0.96	0.72						
D	17.80	+∞	19.44	19.92	15.00					
E	15.80	+∞	17.28	17.52	13.00	0				
F	16.50	+∞	17.76	18.72	13.50	0	0			
G	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞		
H	19.49	+∞	37.40	21.60	25.13	23.63	22.13	15.00	+∞	

(a) The Activity Distance Matrix

AR	A	B1	B2	B3	C	D	E	F	G	H
A	1.00									
B1	1.00	1.00								
B2	0.62	1.00	1.00							
B3	0.49	1.00	0.00	1.00						
C	0.14	1.00	0.38	0.49	1.00					
D	0.00	1.00	0.00	0.00	0.00	1.00				
E	0.00	1.00	0.00	0.00	0.00	0.00	1.00			
F	0.00	1.00	0.00	0.00	0.00	0.00	0.00	1.00		
G	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	
H	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1

(b) The Activity Relatedness Matrix

OR	Mike	Carol	Pam	John	Anne	Wil	Mary	Sara	Pete	Sam
Mike	1									
Carol	0.2698	1								
Pam	0.2482	0.5	1							
John	0.0815	0.25	0.25	1						
Anne	0.1779	0.2585	0.2548	0.1298	1					
Wil	0.3333	0.5	0.5	0	0.25	1				
Mary	0.3333	0.5	0.5	0	0.25	1	1			
Sara	0.1846	0.5	0.3333	0.25	0.1723	0.3333	1	1		
Pete	0.2698	0.5	0.5	0.25	0.2585	0.5	0.5	0.3333	1	
Sam	0.163	0.5	0.5	0.5	0.2595	0	0	0.3333	0.5	1

(c) The Originator Relatedness Metric

Figure 3. Example Pattern Application Metrics

In order to map the *AR* to the relatedness between originators, we need to use the assignment operation π_A defined in Definition 1. According to Definition 1, $\pi_A(o_i) = \{a_i | a_i \in \mathcal{A}, i = p, \dots, q\}$. With the help of the assignment function π_A, we can now define the *Originator Relatedness* (*OR*) as following (Definition 10).

Definition 10: (Originator Relatedness). For any pair of originators $o_i, o_j \in \mathcal{O}$, through the assignment function we can get $\pi_A(o_i) = \{a_i | a_i \in \mathcal{A}, i = p, \dots, q, p < q\}$, $\pi_A(o_j) = \{a_j | a_j \in \mathcal{A}$,

$j = m, \dots, n, m < n\}$. Thus, the *Originator Relatedness* (*OR*) is defined as:

$$OR(o_i, o_j) = \begin{cases} \dfrac{\sum_{k=p}^{q} \sum_{l=m}^{n} AR(a_i, a_j)}{(q - p - 1) \times (n - m - 1)}, if \ o_i \neq o_j \\ 1, if \ o_i = o_j \end{cases}$$

The *OR* metric is transitive/symmetric (e.g. $OR(o_i, o_j) = OR(o_j, o_i)$). Also, in order to achieve more accurate organizational models, a threshold (γ) on a particular activities undertaken by a specific originators is defined (occurrence of the activity by the originator divided by overall occurrence of the

activity) – only *AR*s of the activities greater the threshold will be considered in the calculation of *OR*. For instance, the occurrence of Activity A by originator Mike in the example log is 4/6 = 0.67. Since the sample size of the example log is small (6 instances), we hereby set the threshold to 0.

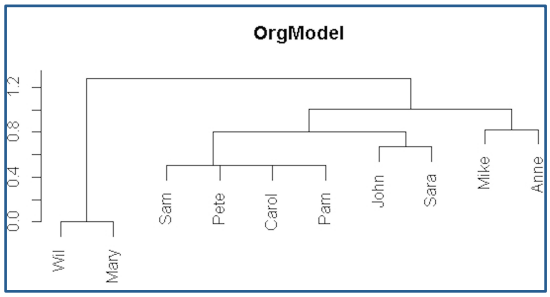
Figure 3(a) shows the *Activity Distance Matrix* between the activities in the example log based on Definition 8. The cells highlighted in pale blue reflect the interleaving patterns between the activities (e.g. (B2, B3)); the cells highlighted in dark blue reflect the exclusiveness patterns between the activities (e.g. (A, B1)); the other cells reflect the strict order patterns. Figure 3(b) shows the *AR* metric between the activities in the example log based on Definition 9. It may be noted that the cells highlighted in dark yellow are rounded to 0. Figure 3(c) shows the *OR* metric between the activities in the example log based on Definition 10.

Algorithm 2: The Organizational Mining Algorithm (Org-AHC)

INPUT: $\mathcal{L}$ -- Event log;
INPUT: $[OR(o_i, o_j)]$ – The matrix of Originator Relatedness between $o_i, o_j, \forall o_i, o_j \in \mathcal{O}$.
OUTPUT: $Dendrogram_k, k \in [1, |\mathcal{O}|]$.

```
1  //Initialize:
2  DEFINE measure; //Define the cluster threshold;
3  DEFINE step; //The constant subtracted from each step, control granularity;
4  GET L; //Read event log;
5  GET [OR(o_i, o_j)];
6  BEGIN
7  C(i) = {o_i}, ∀i, o_i ∈ O;
8  FOR (k = |O| ; k −−; k > 1)
9    FOREACH (o_i, o_j ∈ O)
10     IF (OR(o_i, o_j) > measure) THEN
11       Dendrogram_k = {C(1), … , C(k)};
12       C(i) = JOIN (C(i), C(j)), ∀i, j;
13       DELETE C(j);
14     END IF;
15   END FOREACH;
16   measure = measure + step;
17 END FOR;
18 DISPLAY Dendrogram_k;
19 END;
```

(a) The Org-AHC Algorithm



(b) The Organizational Model from the Example Log

Figure 4. (a) *Org-AHC* Algorithm and (b) Example Organizational Model

Abovementioned metrics can derive a flat organizational model; however, organizational models often follow a hierarchical fashion. To derive hierarchical organizational models, we apply an *Agglomerative Hierarchical Clustering* (AHC) technique adapted from the *group-average AHC* algorithm proposed in (Shepitsen et al. 2008). The proposed *Org-AHC* clustering algorithm is shown in Figure 4(a).

The *Org-AHC* algorithm conducts stepwise clustering on originators when *OR* between two originators are greater than the threshold (*measure*) at level $k \in [1, |\mathcal{O}|]$. It then moves up to the (*k - 1*) level with threshold updated to (*measure + step*). The algorithm halts at (*k = 1*), which implies that all originators belong to the same cluster.

Using the example log shown in Table 1, we can obtain the *Activity Relatedness Matrix* (based on Definition 9) and *OR* matrix (based on Definition 10). By applying the Org-AHC algorithm, the organizational model in the form of a dendrogram is shown in Figure 4(b). It can be noted that {John, Sam, Pete, Sara} are merged into the same organizational unit (or role) since they undertake similar activities in the example log – thus, they are labeled as '*in-house*' reviewers. Similarly, {Carol, Pam} ('*external reviewers*') and {'Anne, Mike'} ('*editors*') can be labeled in the organizational model. The only issue in this organizational model is that originator *Mary* should be grouped with {Carol, Pam} since *Mary* is an external reviewer according to Table 1; although originator *Wil* should be grouped with {Anne, Mike} because he is some kind of decision maker (e.g. '*editor-in-chief*'). The reason causing this issue is that there are some short loops (e.g. E-F-D, E-D) in the example log – thus, the pattern between each pair from Activities *D*, *E*, and *F* are all interleaving. Also, in this small example, *Wil* and *Mary* only conduct one activity respectively ('*Wil*, *D*', '*Mary*, *F*') – which *erroneously* increases their originator relatedness. One approach to counteract this issue is to increase the size of instances so that the originators would possibly have more than one activity. Another way to amend this issue is to develop a method to deal with the short loops in event logs.

A Case Study

For the case study, we selected the complete "review process" event log (c.f. http://data.3tu.nl/repository/uuid:da6aafef-5a86-4769-acf3-04e8ae5ab4fe). The event log contains 10,000 traces (papers) and 236,360 events (in 8 distinct event classes). 10 originators are involved in the event log. The reviewing process is similar to the one discussed in the example log. Since the sample size of the selected event log is big, we set the threshold γ = 0.1.

As discussed earlier, the output of each module is shown in Figure 5.

		Predecessors									
	Task	A	B1	B2	B3	C	D	E	F	G	H
Successors	A	-	-	-	-	-	-	-	-	-	-
	B1	→	-	-	-	-	-	-	-	-	-
	B2	→	||	-	-	-	-	-	-	-	-
	B3	→	||	||	-	-	-	-	-	-	-
	C	→	→	→	→	-	-	-	-	-	-
	D	→	→	→	→	→	-	-	-	-	-
	E	→	→	→	→	→	||	-	-	-	-
	F	→	→	→	→	→	||	||	-	-	-
	G	+	+	+	+	+	+	+	+	-	-
	H	→	→	→	→	→	→	→	→	+	-

(a) Behavioral Relation Matrix

AD	A	B1	B2	B3	C	D	E	F	G	H
A										
B1	0.61									
B2	0.47	0								
B3	0.74	0	0							
C	1.7	0.72	0.91	0.72						
D	7.66	8.79	11.2	15.49	2.24					
E	8.1	11.32	13.76	11.72	11.7	0				
F	10.75	14.49	14.58	16.6	12.5	0	0			
G	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞		
H	17.58	16.5	15.34	17.68	18.14	17.82	9.34	8.24	+∞	

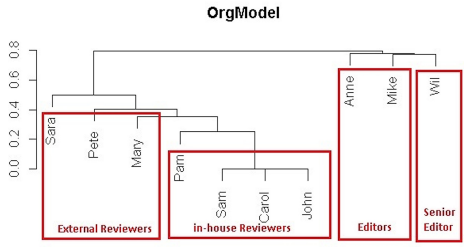
(b) The Activity Distance Matrix

AR	A	B1	B2	B3	C	D	E	F	G	H
A	1									
B1	0.543	1								
B2	0.625	1	1							
B3	0.477	1	1	1						
C	0.183	0.491	0.463	0.549	1					
D	0	0	0	0	0.130	1				
E	0	0	0	0	0	1	1			
F	0	0	0	0	0	1	1	1		
G	0	0	0	0	0	0	0	0	1	
H	0	0	0	0	0	0	0	0	0	1

(c) The Activity Relatedness

OR	Mike	Carol	Pam	John	Anne	Wil	Mary	Sara	Pete	Sam
Mike	1									
Carol	0.2099	1								
Pam	0.2074	0.75	1							
John	0.2099	1	0.75	1						
Anne	0.2146	0.2099	0.2074	0.2099	1					
Wil	0.226	0	0.25	0	0.226	1				
Mary	0.2076	0.6667	0.5833	0.6667	0.2076	0.3333	1			
Sara	0.2026	0.5	0.5	0.5	0.2026	0.5	0.5	1		
Pete	0.1415	0.5833	0.5833	0.5833	0.1415	0.3333	0.5556	0.5	1	
Sam	0.2678	1	0.75	1	0.2678	0	0.6667	0.5	0.6667	1

(d) The Originator Relatedness



(e) Organizational Model

Figure 5. Case Study Artifacts

The outputs shown in Figure 5(a) – (e) use similar notations, comparing to the ones shown in the prior section (Table 2, Table 3, and Figure 3(b)). We can roughly divide Figure 5(e) into three parts. The most left side of Figure 5(e), containing {Sara, Pete, Mary}, consists the '*external reviewers*' group. The middle part of Figure 5(e), containing {Sam, Carol, John, Pam}, consists the '*in-house reviewers*' group. The '*external reviewers*' group and the '*in-house reviewers*' group are merged together as the '*reviewers*' group. The most left side of Figure 5(e), containing {Mike, Anne, Wil}, consists the '*editors*' group. Then the '*reviewers*' group and the '*editors*' group are merged together.

Based on the assumption that no explicit process model and organizational model exists on selected event log, we conducted an empirical evaluation on the discovered organizational model for its accuracy. Four researchers were asked to create the organizational model based on the "review example large" event log. To keep the evaluation unbiased, we did not show the organizational model in Figure 5(e) to the researchers until they created their own results. The results after reconciliation from the researchers are

shown in Figure 6. We use the *Cophenetic Correlation Coefficient* (CCC) (Fowlkes and Mallows 1983) to evaluate the accuracy of the organizational model illustrated in Figure 5(e), using the model in Figure 6 as the "*ground truth*". The CCC values lie in the interval [-1,1] – if the CCC value is close to 1, the two dendrograms are of higher resemblance – meaning the model in Figure 5(e) is more accurate. The CCC value between the two dendrograms is *0.791*, which is closer to 1 than to 0 – which indicates that the organizational model generated by the *OrgMiner* framework is in accordance with the judgments from domain experts.

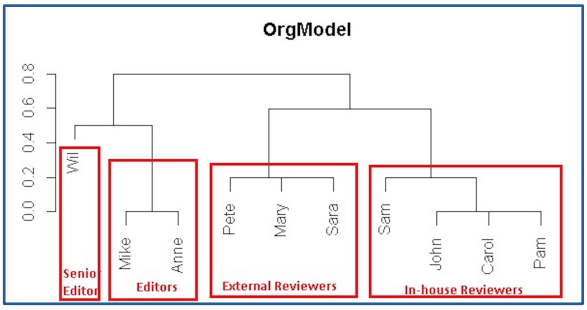


Figure 6. Manually Constructed Organizational Model

Conclusion

We have proposed a behavioral pattern based framework (namely *OrgMiner*) in this paper. We have defined the concept of behavioral patterns, which rely on the weak order relation appearing in event logs. Besides, we quantify the behavioral patterns through different metrics and then apply them for organizational mining purpose. In this way, organizational models clustering originators of activities in event logs can be constructed without explicit process and/or organizational models. Through an empirical evaluation, the organizational models generated by *OrgMiner* are satisfactory for further analytical purposes.

To increase the accuracy of mined organizational models, we need to develop a method to deal with the short loops in event logs and to improve the sensitivity of our approach. Also, we only considered the sequential relations between activities in event logs; in the future, more domain-specific knowledge (e.g. in the form of domain ontologies) might be utilized to derive more complex relations – such domain specific knowledge may be extracted from other organizational knowledge artifacts such as existing process models, business policies, and so forth. Furthermore, we believe the identified behavioral patterns can be applied for other PMA goals, such as social network analysis, decision point mining, mining dataflow and/or data object interactions, performance checking, and cost-benefit analysis. We also acknowledge that, our work highly relies on the availability, meaningfulness, and correctness of the event logs. This implies requiring a pre-processing step in the form of normalizing the event logs for ensuring their meaningfulness, and/or merging event logs from different information systems together, thus preparing them for applying our organizational mining approach on them. These are avenues for future applications of the proposed *OrgMiner* framework.

References

Van der Aalst, W. M. P. 2011. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, New York, NY: Springer, p. 352.

Van der Aalst, W. M. P. 2012. "Process mining: Overview and opportunities," *ACM Transactions on Management Information Systems* (3:2), pp. 1–17.

Van der Aalst, W. M. P., Alves de Medeiros, A. K., and Weijters, A. J. M. M. 2005. "Genetic process mining," in *Applications and Theory of Petri Nets 2005, Proceedings*, G. Ciardo and P. Darondeau (eds.), (Vol. 3536) Berlin: Springer-Verlag, pp. 48–69.

Van der Aalst, W. M. P., and ter Hofstede, A. H. M. 2003. "Workflow Patterns," *Distributed and Parallel Databases* (14:1), pp. 5–51.

Van der Aalst, W. M. P., and Song, M. 2004. "Mining social networks: Uncovering interaction patterns in business processes," in *Proceedings of the International Conference on Business Process Management (BPM 2004)*, B. Pernici and M. Weske (eds.), (Vol. 3080) , pp. 244–260.

Van der Aalst, W. M. P., Weijters, T., and Maruster, L. 2004. "Workflow mining: Discovering process models from event logs," *IEEE Transactions on Knowledge and Data Engineering* (16:9), pp. 1128–1142.

Agrawal, R., and Srikant, R. 1994. "Fast Algorithms for Mining Association Rules," in *Proceeding VLDB '94 Proceedings of the 20th International Conference on Very Large Data Bases*, , pp. 487–499.

Bae, J., Caverlee, J., Liu, L., and Yan, H. 2006. "Process mining by measuring process block similarity," in *Business Process Management Workshops*, (Vol. 4103) Berlin: Springer-Verlag Berlin, pp. 141–152.

Bertolini, M., Bevilacqua, M., Ciarapica, F. E., and Giacchetta, G. 2011. "Business process re-engineering in healthcare management: a case study," *Business Process Management Journal* (17:1), pp. 42–66.

Bose, R., and van der Aalst, W. M. P. 2010. "Trace clustering based on conserved patterns: Towards achieving better process models," *Business Process Management Workshops* .

Bose, R., and van der Aalst, W. M. P. 2012. "Process Diagnostics Using Trace Alignment : Opportunities , Issues , and Challenges," *Information Systems* (37:2), pp. 117–141.

Bucher, T., Gericke, A., and Sigg, S. 2009. "Process-centric business intelligence," *Business Process Management Journal* (15:3), pp. 408–429.

Van Dongen, B. F., and van der Aalst, W. M. P. 2004. "EMiT: A process mining tool," in *Applications and Theory of Petri Nets 2004, Proceedings*, (Vol. 3099) Berlin: Springer-Verlag Berlin, pp. 454–463.

Dumas, M., van der Aalst, W. M. P., and ter Hofstede, A. H. M. (Eds.). 2005. *Process Aware Information Systems: Bridging People and Software through Process Technology*, Hoboken, NJ: John Wiley & Sons, Inc.

Ferreira, D. R., and Thom, L. H. 2012. "A Semantic Approach to the Discovery of Workflow Activity Patterns in Event Logs," *International Journal of Business Process Integration and Management* (6:1), pp. 4–17.

Fowlkes, E. B., and Mallows, C. L. 1983. "A Method for Comparing Two Hierarchical Clusterings," *Journal of the American Statistical Association* (78:383), pp. 553–569.

Günther, C., and van der Aalst, W. 2007. "Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics," in *Business Process Management: 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007 Proceedings*, , pp. 328–343.

IEEE Task Force on Process Mining. 2011. "Process Mining Manifesto," .

Jareevongpiboon, W., and Janecek, P. 2013. "Ontological approach to enhance results of business process mining and analysis," *Business Process Management Journal* (19:3), pp. 459–476.

Köck, M., and Paramythis, A. 2011. "Activity sequence modelling and dynamic clustering for personalized e-learning," *User Modeling and User-Adapted Interaction* (21:1-2), pp. 51–97.

Pérez-Castillo, R., Weber, B., Pinggera, J., Zugal, S., de Guzmán, I. G.-R., and Piattini, M. 2011. "Generating event logs from non-process-aware systems enabling business process mining," *Enterprise Information Systems* (5:3), pp. 301–335.

Savasere, A., Omiecinski, E., and Navathe, S. 1995. "An efficient algorithm for mining association rules in large databases," .

Sellami, R., Gaaloul, W., and Defude, B. 2013. "Process Socio Space Discovery Based on Semantic Logs," *Journal of Internet Technology* (14:3), pp. 401–412.

Shepitsen, A., Gemmell, J., Mobasher, B., and Burke, R. 2008. "Personalized recommendation in social tagging systems using hierarchical clustering," *Proceedings of the 2008 ACM conference on Recommender systems - RecSys '08* New York, New York, USA: ACM Press, p. 259.

Smirnov, S., and Weidlich, M. 2009. "Action patterns in business process models," *Service-Oriented Computing, Lecture Notes in Computer Science* (5900/2009), pp. 115–129.

Smirnov, S., Weidlich, M., Mendling, J., and Weske, M. 2012. "Action Patterns in Business Process Model Repositories," *Computers in Industry* (63:2), pp. 98–111.

Song, M., and van der Aalst, W. M. P. 2008. "Towards comprehensive support for organizational mining," *Decision Support Systems* (46:1)Elsevier B.V., pp. 300–317.

Song, M., Günther, C. W., and van der Aalst, W. M. P. 2009. "Trace clustering in process mining," *BPM Workshops, Lecture Notes in Business Information Processing* (17:2), pp. 109–120.

Sun, S. X., and Zhao, J. L. 2013. "Formal workflow design analytics using data flow modeling," *Decision Support Systems* (55:1), pp. 270–283.

Thomas, O., and Fellmann, M. 2006. "Semantic event-driven process chains," in *Proceedings of the Workshop on Semantics for Business Process Management (SBPM '06), held at the 3rd European Semantic Web Conference (ESWC 2006)*, Budva, Montenegro, June.

Tiwari, A., Turner, C. J., and Majeed, B. 2008. "A review of business process mining: state-of-the-art and future trends," *Business Process Management Journal* (14:1), pp. 5–22.

Veiga, G. M., and Ferreira, D. R. 2010. "Understanding Spaghetti Models with Sequence Clustering for ProM," *Business Process Management Workshops - Lecture Notes in Business Information Processing* (43), pp. 92–103.

Weidlich, M., Mendling, J., and Weske, M. 2011. "Efficient Consistency Measurement Based on Behavioral Profiles of Process Models," *Software Engineering, IEEE Transactions on* (37:3), pp. 410–429.

Weske, M. 2012. *Business Process Management: Concepts, Languages, Architectures*, (2nd ed.) , p. 403.