OrdinoR: A Framework for Discovering, Evaluating, and Analyzing Organizational Models using Event Logs

Jing Yang[a,*], Chun Ouyang[a], Wil M.P. van der Aalst[b,a], Arthur H.M. ter Hofstede[a], Yang Yu[c]

[a]*Queensland University of Technology, Australia*
[b]*RWTH Aachen University, Germany*
[c]*Sun Yat-sen University, China*

Abstract

In order to streamline business processes and increase competitiveness, organizations need to have a deep insight into the resources that they deploy. Among others, they need to understand how these resources act in groups to achieve organizational outcomes. Accurate and timely information is a sine qua non to achieve this understanding. Process mining can be exploited for the purpose of deriving organizational models from event logs that contain resource-related data. But existing process mining techniques are not fully up to this task, as they are not able to cope with the multi-faceted nature of business processes and are not yet able to determine how resource groupings are involved in process execution. In addition, there is no provision for how to evaluate the quality of discovered organizational models.

To tackle these challenges, we propose a novel framework, *OrdinoR*, capable of supporting discovery of organizational models using event logs, their evaluation, and their analysis. *OrdinoR* is constructed around a rich notion of organizational model where resource groupings are linked to multiple dimensions of process execution. The framework also provides a set of measures for systematically evaluating such models and analyzing the behavior of resource groups therein. Experiments have been conducted to evaluate the framework using a publicly available real-life dataset. These demonstrate the usefulness of the approach.

Keywords: event log, organizational model, process mining, conformance checking

*Corresponding author

Email addresses: `roy.j.yang@qut.edu.au` (Jing Yang), `c.ouyang@qut.edu.au` (Chun Ouyang), `wvdaalst@pads.rwth-aachen.de` (Wil M.P. van der Aalst), `a.terhofstede@qut.edu.au` (Arthur H.M. ter Hofstede), `yuy@mail.sysu.edu.cn` (Yang Yu)

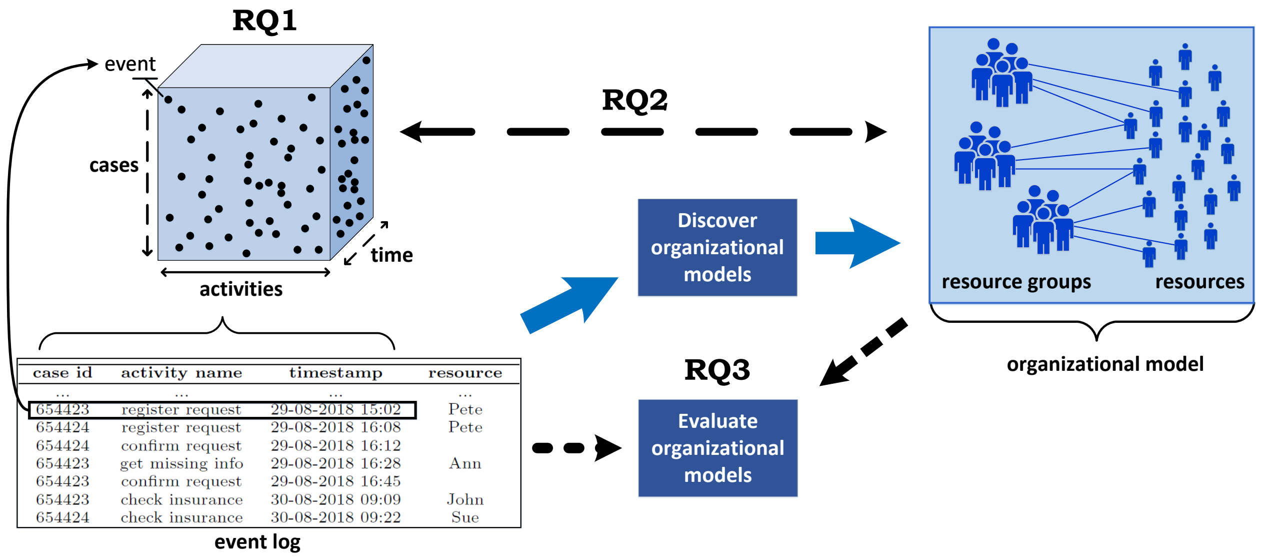
1. Introduction

Modern organizations operate in an ever-changing context. To do so successfully, they have to be able to rapidly adapt their business processes and optimally marshal their resources [1]. This is easier to achieve when these processes are formally captured and executed by business process management systems and the human resources are members of appropriate organizational groupings. The capability to constantly evolve organizational groups alongside changing business processes is essential for organizations [1, 2] and it is thus imperative that they maintain accurate and timely insights into these groups [3]. Clearly, relying on organizational charts — too static and often too high-level — or on managers' intuition — too vague and often anecdotal — will not be conducive to achieving this capability [4].

A promising approach to obtaining accurate and timely insights into resource groupings and their involvement in business processes is through the use of process execution data, which is readily available in many contemporary process-aware information systems [5], e.g., Enterprise Resource Planning systems. This data is stored in so-called event logs and records *activities* undertaken at a specific *time* in the context of the execution of a certain instance of a process (often known as a *case*) [5, 1]. In addition, it may record *resources* who executed those activities. As such, event logs capture the trails of human resource participation in actual business process execution, and therefore provide a reliable starting point for mining timely process- and resource-related information [6, 7].

Process mining [5] offers a growing body of methods to extract knowledge from event logs for process management and improvement. The subfield of *organizational model mining* [6] is concerned with the study of groups of human resources, specifically how models can be derived from event logs to reflect resource groupings in process execution. The relatively underexplored area of organizational model mining [6, 8, 9] contains some research gaps which impede its use in practice. Three open issues in particular will be explored in this paper.

Figure 1 illustrates these issues. Process execution concerns three primary dimensions, i.e., case, activity, and time. Thus, events in a log can be viewed as data points in three-dimensional space. Existing methods for mining organizational models mainly focus on the activity dimension, but rarely consider the case and time dimensions. This narrow focus is limiting when resource groupings need to be considered across different cases (e.g., specialist groups dedicated to particular customers) or across different time periods (e.g., employees playing the same role but working



case id	activity name	timestamp	resource
...	...	...	...
654423	register request	29-08-2018 15:02	Pete
654424	register request	29-08-2018 16:08	Pete
654424	confirm request	29-08-2018 16:12	
654423	get missing info	29-08-2018 16:28	Ann
654423	confirm request	29-08-2018 16:45	
654423	check insurance	30-08-2018 09:09	John
654424	check insurance	30-08-2018 09:22	Sue

Figure 1: An illustration of the research gaps in organizational model mining (represented by the dashed lines)

different shifts). Furthermore, organizational models discovered by existing methods often do not transcend the mere clustering of resources — they do not describe how the discovered resource groups were involved in process execution. It is therefore challenging to use these models for understanding the behavior of resource groups. Last but not least, existing research relies on either domain knowledge or mining-method specific metrics to evaluate discovered models — a generic evaluation approach is still missing. In view of these gaps, we formulate three research questions:

RQ1. How to utilize information on multiple dimensions of process execution when discovering organizational models from an event log?

RQ2. How to link discovered resource groups with process execution to describe their involvement?

RQ3. How to evaluate discovered organizational models against input event logs?

In this paper, we address these research questions by proposing a novel framework, namely *OrdinoR*[1], for organizational model mining from event logs. Our contributions are as follows.

- We define a new, rich notion of organizational model as the foundation of the framework. It considers multiple dimensions of process execution and links the relevant execution information with resource groupings (addressing RQ1 and RQ2). Such a model can capture comprehensive knowledge about resource groups and their involvement in processes (which

[1] "Ordino" means "to arrange" in Latin, and the *R* stands for "resources".

has not been addressed in the literature).

- We propose two measures, fitness and precision, for assessing discovered organizational models against input event logs. These measures form a generic basis for model evaluation in the framework (addressing RQ3).

- We propose a set of measures for analyzing organizational models, whose application deepens the understanding of how resource groups and their members are involved in process execution (a further contribution to RQ2).

- We operationalize our framework by developing an approach to discover, evaluate, and analyze organizational models, and by implementing an open-source tool.

- We conduct extensive and replicable experiments on publicly available, real-life event logs to test our proposals and report our findings.

Our research extends the process mining literature on knowledge discovery of the organizational perspective by identifying key research gaps and proposing a novel approach to address them. Application of our research empowers organizations by guiding decisions on organizational structure design and staff deployment toward process improvement.

The remainder of this paper is structured as follows. Section 2 provides background to our research and related work. Section 3 introduces the research method applied in our research. Section 4 elaborates on the design of the proposed framework *OrdinoR*. Section 5 presents the approach as a realization of this framework. Section 6 reports on experiments conducted on real-life event log data along with our findings. Section 7 discusses the limitations of our work and directions for future investigations. Finally, Section 8 concludes the paper.

2. Background and Related Work

A business process consists of a set of logically connected *activities* performed in an organization and captures possible alternative ways to achieve a business goal [2, 1]. An instance of the execution of a process is a *case* [5]. Process execution involves human *resources* performing a sequence of activities in the process. Human resources play roles and hold positions in an organization, and form *resource groups* [1, 6] representing various organizational entities, such as departments and project teams. Data related to process execution is recorded by process-aware information systems, notably in the form of *event logs* [5]. An event log consists of a set of events with a range of event attributes, providing factual information on how activities were performed.

Process mining can be used to derive insights from event logs, helping managers of organizations understand and improve human resource management in business processes [6, 7]. There are four main research topics in process mining relevant to human resources, including organizational model mining (e.g., [6, 10, 9]), social network mining (e.g., [11, 12, 13]), rule mining (e.g., [7, 14, 15]), and behavior profile mining (e.g., [16, 4, 17]).

The focus of this paper is on organizational model mining. As in general process mining research, the key problem in organizational model mining concerns the discovery of models from input event logs to represent the real behavior of process execution [5] — in this case, the groupings of resources [6]. State-of-the-art research (e.g., [8, 9, 18, 19]) addresses the problem by first characterizing how individual resources participate in process execution or how they interact with each other. Then, the problem is transformed into a clustering (as in [18, 9]) or a community-detection problem (as in [8, 19]), and dedicated techniques are applied to solve them. As a result, a discovered organizational model comprises groups of resources with similar characteristics in process execution.

Guided by our research questions, we establish three perspectives to review the related work. The first perspective concerns the dimensions of process execution considered when discovering organizational models. An event log suitable for mining organizational models records information on activity labels, resource identifiers, case identifiers, and timestamps. Hence, the participation of human resources in process execution can be analyzed from multiple dimensions — how resources carry out activities (*activity*), how they are involved in different cases (*case*), how they work at different times (*time*), and how they interact with each other (*resource interaction*). The second perspective concerns whether discovered models capture the involvement of resource groups in process execution. The third perspective concerns the evaluation of discovered models. Among the related work, some merely demonstrate the use of their proposed methods (*demonstration only*), while others evaluate the quality of discovered models by either comparing a discovered model against relevant domain knowledge (*compare to domain knowledge*), e.g., official organizational structures, or adopting measures to assess the effectiveness of the applied technique (*assess performance*). Since the model discovery problem targets deriving a model from event logs to capture resource groupings in process execution, we also consider input event logs necessary for evaluating discovered models (*refer to input log*). This is consistent with how model quality is typically evaluated in process mining research [1, 5], by comparing model behavior to log data.

Table 1 classifies the existing research on organizational model mining in terms of these three perspectives. Below, we elaborate on the analysis of literature.

Table 1: Classification of state-of-the-art literature on discovering organizational models from event logs

Paper	Dimensions considered in model discovery				Discovered models capturing resource group involvement	Evaluation of discovered models			
	activity	case	time	resource interaction		demonstration only	compare to domain knowledge	assess performance	refer to input log
[6]	√	√			√		√		
[10]	√			√	√		√		
[18]	√						√		
[20]	√				√	√			
[21]	√			√	√	√			
[19]	√							√	
[22]	√					√			
[8]	√							√	
[9]	√						√		
[23]	√						√		
[24]				√		√			
[25]				√		√			
[26]				√		√			

1. Are the multiple dimensions of process execution considered when discovering organizational models? Most existing methods only consider the activity dimension, since common resource grouping schemes (e.g., business roles, functional units) often result in specialized groups of employees handling specific activities in a process. Some existing research exploits the information on resource interactions (e.g., handover of work between the execution of adjacent activities), in particular studies that focus on the reporting relationships among employees [24, 25, 26]. On the other hand, information related to cases and time is rarely considered when discovering organizational models. Only Song and Van der Aalst [6] exploit case information in event logs and discover employee teams assembled for collective tasks. The multiple dimensions of process execution are yet to be explored for discovering various resource groupings.

2. Do the discovered organizational models describe the involvement of groups in process execution? In existing work, discovered organizational models can represent groupings of human resources. But only a few approaches [6, 20, 21, 10] allow their models to capture the connection between identified resource groups and process execution. This connection should characterize the precise involvement of these groups in the process, e.g., their responsibilities or permissions. Establishing this connection is important to understanding how resources are best deployed.

3. How are the discovered models evaluated? Some existing studies only concern a demonstration of how a proposed method can be applied to an event log to discover organizational models [20, 21, 22, 24, 25, 26], without measuring the quality of the discovered models. Others conduct evaluation on the discovered models by two means. The first is to compare models with some domain knowledge that is relevant to resource groupings in the process, such as official organizational structures or business roles [6, 18, 10, 9, 23]. Clearly, this relies on the availability of domain knowledge. In addition, the evaluation results can be flawed, since human resource groupings in reality may deviate from the referenced domain knowledge. Another means is to assess the performance of the applied techniques for model discovery [8, 19], which often requires using a technique-specific method. For example, Appice [8] applies a community-detection technique to discover organizational models and adopts a measure named "modularity" to evaluate how effectively that community-detection technique performs. However, modularity is a measure specific to community-detection problems and cannot be applied to evaluate models discovered using other techniques, e.g., those based on cluster analysis. So far, none of the existing studies has considered using input event logs for evaluating the quality of organizational models discovered from the logs.

The remainder of this paper introduces how we address the foregoing issues by establishing a novel framework for organizational model mining.

3. Research Method

We adapt and apply the Design Science Research Methodology (DSRM) [27] and follow the guidelines proposed by Hevner et al. [28] to develop our framework — a purposeful and viable artifact for organizational model mining in the process mining domain (Guideline 1: Design as an Artifact). Figure 2 illustrates our research method.

Existing research [6, 29, 7] has shown the need to support the understanding of human resource behavior in business processes and the value of exploiting process execution data for that purpose (Guideline 2: Problem Relevance). We start the DSRM process by reviewing the literature and identifying the key open issues in state-of-the-art organizational model mining research, introduced in the previous section.

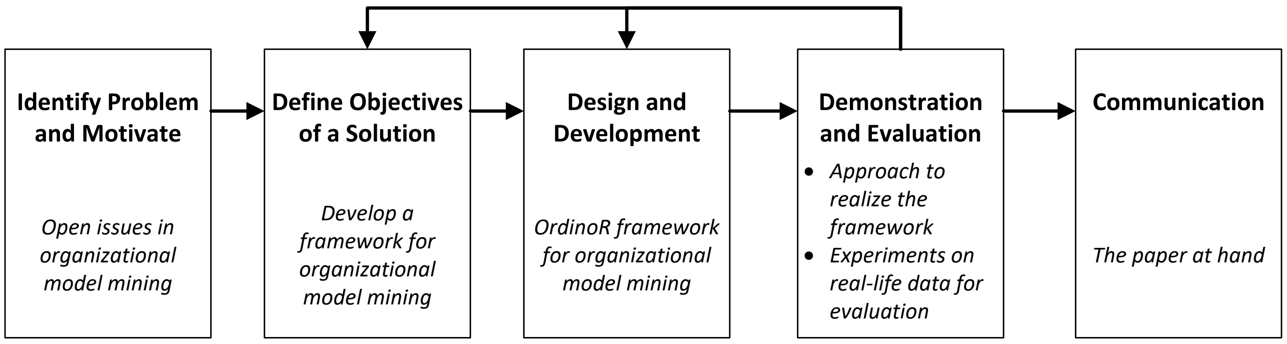

Identify Problem and Motivate
Open issues in organizational model mining
Define Objectives of a Solution
Develop a framework for organizational model mining
Design and Development
OrdinoR framework for organizational model mining
Demonstration and Evaluation
Approach to realize the framework
Experiments on real-life data for evaluation
Communication
The paper at hand

Figure 2: Apply DSRM [27] to develop the OrdinoR framework for organizational model mining

In view of the identified gaps, we define our main objective as to develop a framework that satisfies several requirements, including: (1) it should enable discovering organizational models from event logs by considering multi-dimensional process execution information; (2) discovered organizational models in this framework should describe how resource groups were involved in process execution; and (3) discovered models should be evaluated against input event logs. Based on these requirements, we rigorously define and formalize the notions related to event logs and organizational models, and use them as the basis for designing and developing our framework (Guideline 5: Research Rigor). Section 4 elaborates on the framework.

We propose an approach to realize the framework and implement it as a software tool (Guideline 5: Research Rigor). Section 5 presents the approach and the implementation, which are then evaluated through experiments on publicly available, real-life data (Guideline 3: Design Evaluation; Guideline 5: Research Rigor) collected from two different business domains. We exploit evaluation results on the collected dataset to iterate the design of our framework and its implementation (Guideline 6: Design as a Search Process). Section 6 reports on the experiments.

Finally, we document the steps taken to develop the framework and share the software implementation as an open-source tool (Guideline 4: Research Contributions). We report our research and its contributions in the paper at hand (Guideline 7: Communication of Research).

4. *OrdinoR* Framework

4.1. Overview

We propose *OrdinoR*, a novel framework for organizational model mining, as shown in Figure 3. We introduce a new, richer notion of organizational model as the foundation of the framework.

Compared with the state-of-the-art, our organizational model specifies not only resources and their groups, but also the connection between resource groups and the multiple dimensions of process execution, captured by the so-called *execution contexts*. We elaborate on these concepts in Sections 4.3 and 4.4.

Built upon the new notion of organizational model, the *OrdinoR* framework is designed to support the following three types of organizational model mining tasks.

1. *Discovery* of organizational models: This task is to construct models from event logs to reflect the groupings of resources and their involvement in process execution (Section 4.5).
2. *Evaluation* of organizational models: This task is to assess model quality by comparing models against event logs using fitness and precision measures (Section 4.6).
3. *Analysis* of organizational models: This task is to examine the actual behavior of resource groups captured in organizational models using event logs. Findings from such analyses can provide insights into group-oriented organizational analytics (Section 4.7).

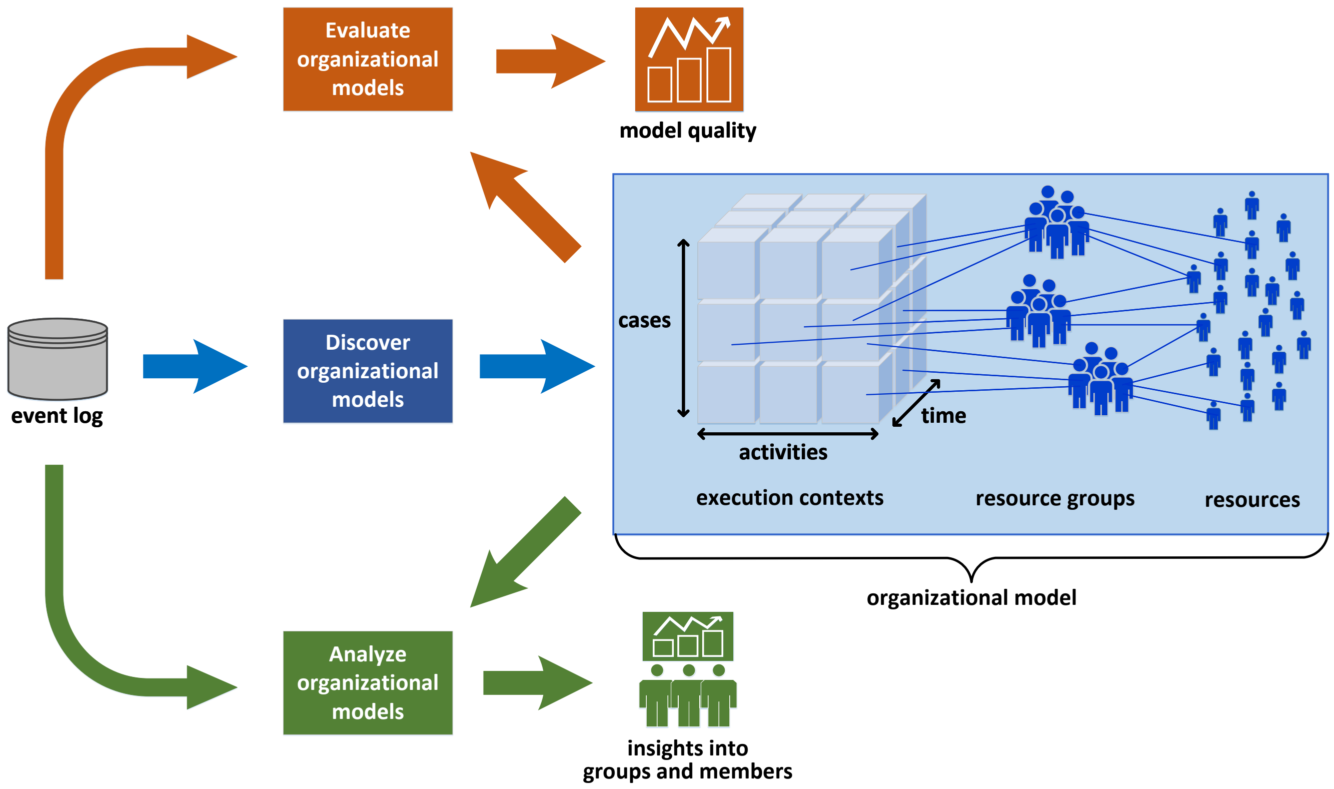

Evaluate organizational models
model quality
event log
Discover organizational models
cases
time
activities
execution contexts
resource groups
resources
organizational model
Analyze organizational models
insights into groups and members

Figure 3: Overview of the *OrdinoR* framework for organizational model mining. Three types of mining tasks are supported: discovery, evaluation, and analysis, highlighted in different colors respectively

4.2. Event Log

The starting point is an event log that records process execution data. Table 2 shows an example event log that records an insurance claim handling process. Rows correspond to events and columns correspond to various event attributes.

Table 2: A fragment of an example event log

case id	activity name	timestamp	resource	customer type
...	...	...	...	...
654423	register request	29-08-2018 15:02	Pete	normal
654424	register request	29-08-2018 16:08	Pete	normal
654424	confirm request	29-08-2018 16:12		normal
654423	get missing info	29-08-2018 16:28	Ann	normal
654423	confirm request	29-08-2018 16:45		normal
654423	check insurance	30-08-2018 09:09	John	normal
654424	check insurance	30-08-2018 09:22	Sue	normal
654425	register request	30-08-2018 10:07	Bob	VIP
654423	accept claim	30-08-2018 11:32	John	normal
654424	reject claim	30-08-2018 11:45	Sue	normal
654423	pay claim	30-08-2018 11:48		normal
654425	confirm request	30-08-2018 12:44		VIP
654425	check insurance	30-08-2018 13:32	Mary	VIP
654425	accept claim	30-08-2018 14:09	Mary	VIP
654425	pay claim	30-08-2018 14:14		VIP
...	...	...	...	...

We define a general data structure for event logs (Def. 1). An event log (EL) contains a set of uniquely identifiable events (E), a set of event attribute names (Att), and the corresponding event attribute values carried by each event (as specified by function π). It is possible that an event does not carry any value for a given event attribute, e.g., in Table 2 there are events with no resource information. Hence, function π is a partial function mapping the attributes of events to values.

Definition 1 (Event Log)**.** *Let $\mathcal{E}$ be the universe of event identifiers, $\mathcal{U}_{Att}$ be the universe of possible attribute names, and $\mathcal{U}_{Val}$ be the universe of possible attribute values. $EL = (E, Att, \pi)$ with $E \subseteq \mathcal{E}$, $E \neq \varnothing$, $Att \subseteq \mathcal{U}_{Att}$, and $\pi \colon E \to (Att \not\rightarrow \mathcal{U}_{Val})$ is an event log. Event $e \in E$ has attributes $dom(\pi(e))$. For an attribute $x \in dom(\pi(e))$, $\pi_x(e) = \pi(e)(x)$ is the attribute value of x for event e.*

Next, we elaborate on the definition of event attributes needed for storing the essential information about process execution (Def. 2). An event log usually records multiple cases. Each case can be uniquely identified and is related to a sequence of events corresponding to activities executed at some specific time. As the minimum requirement for event logs, events have three standard attributes: case identifier (*case*), activity name (*act*), and timestamp (*time*). Optionally, an event records the resource (*res*) executing the activity. In addition to these four common attributes, an event log may record event attributes such as *customer type* and *cost*, which vary across different processes and information systems.

Definition 2 (Event Attributes)**.** *Let $\mathcal{C} \subseteq \mathcal{U}_{Val}$, $\mathcal{A} \subseteq \mathcal{U}_{Val}$, $\mathcal{T} \subseteq \mathcal{U}_{Val}$ and $\mathcal{R} \subseteq \mathcal{U}_{Val}$ denote the universe of case identifiers, the universe of activity names, the universe of timestamps, and the universe of resource identifiers, respectively. Any event log $EL = (E, Att, \pi)$ has four special attributes: $\{case, act, time, res\} \subseteq Att$ such that, for any $e \in E$:*

- $\{case, act, time\} \subseteq dom(\pi(e))$,
- $\pi_{case}(e) \in \mathcal{C}$ *is the case to which e belongs,*
- $\pi_{act}(e) \in \mathcal{A}$ *is the activity e refers to,*
- $\pi_{time}(e) \in \mathcal{T}$ *is the time at which e occurred, and*
- $\pi_{res}(e) \in \mathcal{R}$ *is the resource that executed e if $res \in dom(\pi(e))$.*

Let $E_{res} = \{\, e \in E \mid res \in dom(\pi(e)) \,\}$ denote the set with all events that have resource information and $E_{nres} = \{e \in E \mid res \notin dom(\pi(e))\}$ the set of events that do not have this information.

4.3. Execution Context

A key feature of the organizational models proposed in our research is its ability to capture the involvement of resource groups in process execution. This is achieved through the notion of *execution context*.

In business process execution, the groupings of resources are often associated with certain contexts, as reflected in event logs by the different types of activities or cases performed by resources, or the times when resources perform activities [6]. Consider the example event log of an insurance claim handling process in Table 2. Pete and Bob only performed activity "register (a claim) request", while John, Sue, and Mary performed "check insurance" and decided whether to "accept" or "reject" a claim. This may be linked with different business roles of employees. In the meantime, Bob and Mary only handled a claim from a "VIP" customer, while others only handled claims from "normal" customers. This can be a result of the insurance company setting up separate teams for VIP and normal customers.

We introduce the concepts of *case types*, *activity types*, and *time types* (Def. 3) to categorize the possible values carried by the three standard event attributes in an event log (i.e., *case*, *act*, and *time*), respectively. These are inspired by the research on process cubes [30]. Case types correspond to the classification of cases and can be derived based on relevant case-level event attributes. For instance, in the above example, case types may be determined by the customer types of insurance claims. Similarly, activity types categorize activity names into groups of relevant activities (e.g., registration, approval) and time types classify timestamps into periods (e.g., days of a week, different hours of a day).

Definition 3 (Case Types, Activity Types, and Time Types)**.** *Let $\mathcal{CT}$, $\mathcal{AT}$, and $\mathcal{TT}$ denote the sets of names of case types, activity types, and time types, respectively. The functions $\varphi_{case} \colon \mathcal{CT} \to \mathcal{P}(\mathcal{C})$, $\varphi_{act} \colon \mathcal{AT} \to \mathcal{P}(\mathcal{A})$, and $\varphi_{time} \colon \mathcal{TT} \to \mathcal{P}(\mathcal{T})$ define partitions over $\mathcal{C}$, $\mathcal{A}$, and $\mathcal{T}$, respectively. We will use a special type $\perp$ that is associated with all cases, all activities, and all times, i.e. $\varphi_{case}(\perp) = \mathcal{C}$, $\varphi_{act}(\perp) = \mathcal{A}$, and $\varphi_{time}(\perp) = \mathcal{T}$. The sets $\mathcal{CT}$, $\mathcal{AT}$, and $\mathcal{TT}$ only share this special type and are otherwise mutually disjoint. We define $\mathcal{CO} = \mathcal{CT} \times \mathcal{AT} \times \mathcal{TT}$.*

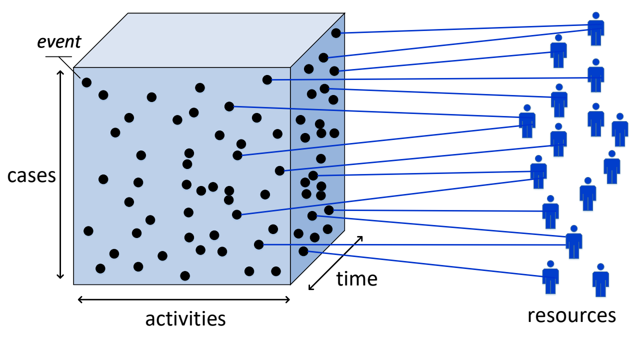
We now formalize the notion of *execution context* (Def. 4) as consisting of a case type, an activity type, and a time type. Each execution context characterizes a possible way of executing an activity in a process and can be associated with a specific set of events that share similar characteristics. For instance, we can relate the first two events in the example log (Table 2) to the same execution context ("normal case", "registration activity", "Wednesday afternoon"). Note that an execution context can be specified with a "wild-card" for any of its constituent components of case type, activity type, and time type. The $\perp$ symbol (formally, as per the previous definition, $\perp$ is a case type, an activity type, and a time type) is used for those components that are not meant to be restricting. Consider for example the execution context ("normal case", $\perp$, "Wednesday"). This execution context concerns all process activities that are executed on Wednesdays when handling insurance claims from normal customers.

Definition 4 (Execution Context)**.** *An execution context co is an element of $\mathcal{CO}$. Given an event log $EL = (E, Att, \pi)$ and an execution context $co = (ct, at, tt)$,*

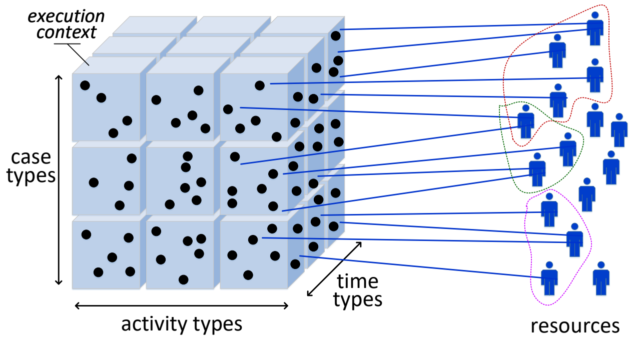
$$[E]_{co} = \{\, e \in E \mid \pi_{case}(e) \in \varphi_{case}(ct) \wedge \pi_{act}(e) \in \varphi_{act}(at) \wedge \pi_{time}(e) \in \varphi_{time}(tt) \,\}$$

is the set of events in the log having that execution context.

Figure 4 illustrates the notion of execution context. In Figure 4a, events are seen as data points in a three-dimensional space capturing information on cases, activities, and time. An event may be related to an individual resource executing an activity. In Figure 4b, event attribute values on each of the three dimensions are partitioned by some specified collection of case types, activity types, and time types. Each combination of a case type, an activity type, and a time type specifies an execution context, represented as a "cube" in the data space. Resources who originated events from the same cubes may belong to the same resource group.



(a) events



(b) execution contexts

Figure 4: Illustration of (a) events as data points in three-dimensional space along the dimensions of case, activity, and time, and (b) execution contexts as "cubes" characterized by case types, activity types, and time types

4.4. Organizational Model

Our notion of *organizational model* (Def. 5) incorporates the concept of execution context. While this model contains, as per usual, the resource groups (RG) and their members (mem), it further captures the involvement of resource groups in process execution, i.e., the "capabilities" of groups, by linking groups with execution contexts (cap).

Definition 5 (Organizational Model). *Let $\mathcal{R}$ be the universe of resources, then $OM = (RG, mem, cap)$ is an organizational model where RG is a set of resource groups, $mem\colon RG \to \mathcal{P}(\mathcal{R})$ maps each resource group onto its members, and $cap\colon RG \to \mathcal{P}(\mathcal{CO})$ maps each resource group onto its possible execution contexts.*

Figure 5 illustrates the proposed notion of organizational model. The many-to-many relationships capture the fact that a resource may belong to multiple groups and a resource group may be associated with multiple execution contexts. Formally, there may exist two distinct groups rg_1 and rg_2 such that $mem(rg_1) \cap mem(rg_2) \neq \varnothing$ and $cap(rg_1) \cap cap(rg_2) \neq \varnothing$.

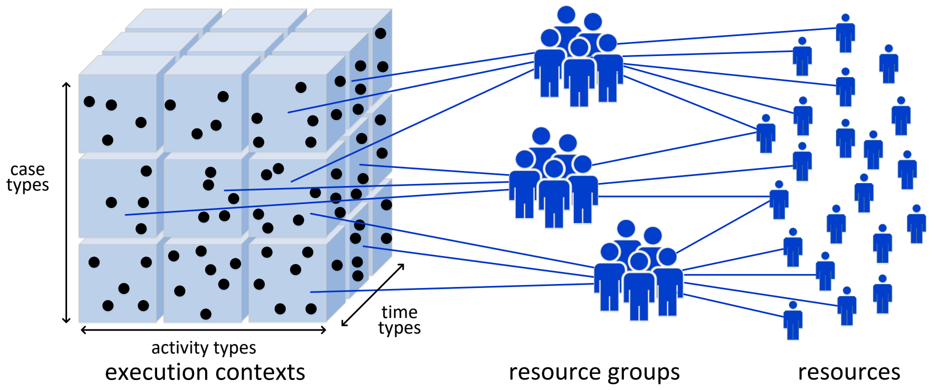


Figure 5: Illustration of an organizational model which illustrates many-to-many relationships between resource groups and resources and those between resource groups and execution contexts

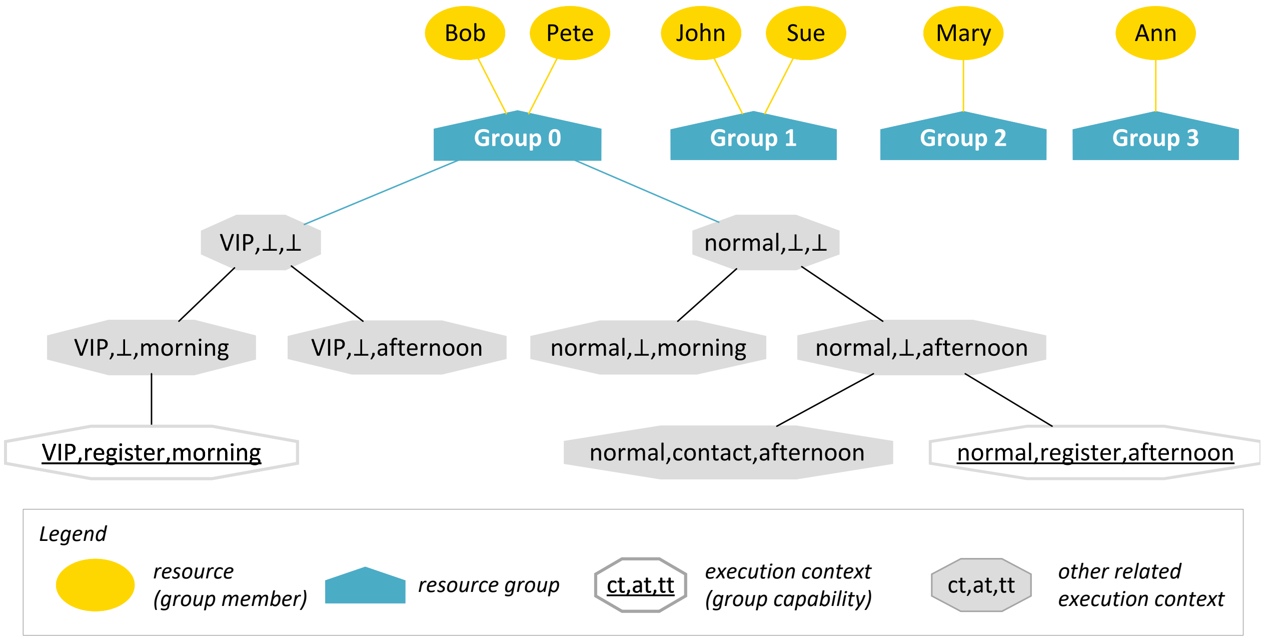


Figure 6: Visualization of an example organizational model related to the event log in Table 2

Figure 6 depicts the visualization of an example organizational model, in which different colored-shapes represent resources, resource groups, and their related execution contexts, respectively. For instance, "Group 0" has two member resources, Bob and Pete, who are capable of executing activities related to execution contexts (VIP, register, morning) and (normal, register, afternoon). These two execution contexts, as the group's capabilities, are underlined in the visualization.

4.5. Discovering Organizational Models

To discover organizational models from an event log, it is useful to view events as samples of resource behavior in process execution [14]. We use the term *resource event* to denote an event that captures a resource's involvement in some execution context. A *resource-event log* (Def. 6) is a multiset of resource events and represents a resource view on process execution data through execution contexts, i.e., a resource-event log records how some resources performed certain activities for certain cases at certain times when they participated in the execution of a process. A resource-event log can be derived from an event log using a collection of execution contexts (Def. 7).

Definition 6 (Resource-Event Log)**.** *A resource event is a tuple* $(r, co) \in \mathcal{R} \times \mathcal{CO}$*. A resource-event log* $RL \in \mathcal{B}(\mathcal{R} \times \mathcal{CO})$ *is a multiset of resource events.*

Definition 7 (Derived Resource-Event Log)**.** *Let* $EL = (E, Att, \pi)$ *be an event log and* $CO \subseteq \mathcal{CO}$ *be a pre-defined collection of execution contexts. The resource-event log derived from EL and CO is* $RL(EL, CO) = [\,(\pi_{res}(e), co) \mid co \in CO, e \in E_{res} \bullet e \in [E]_{co}\,]$.

Table 3 shows an example resource-event log derived from the event log in Table 2, using execution contexts defined based on the case types, activity types, and time types below.

- Two case types are defined based on the event attribute "customer type", which distinguishes two groups of customers, namely *normal* and *VIP*. Therefore, $\{654423, 654424\} \subseteq \varphi_{case}(\text{normal})$ and $\{654425\} \subseteq \varphi_{case}(\text{VIP})$.

- Four activity types are defined: *register*, *contact*, *check*, and *decide*. {"register request", "confirm request"} $\subseteq \varphi_{act}(\text{register})$, {"get missing info", "pay claim"} $\subseteq \varphi_{act}(\text{contact})$, {"check insurance"} $\subseteq \varphi_{act}(\text{check})$, {"accept claim", "reject claim"} $\subseteq \varphi_{act}(\text{decide})$;

- Two time types are defined by dividing working hours in a day into two time frames, namely *morning* and *afternoon*. Therefore, timestamps of events are categorized accordingly, e.g., "30-08-2018 09:09" $\in \varphi_{time}(\text{morning})$, "29-08-2018 15:02" $\in \varphi_{time}(\text{afternoon})$.

Note that a resource event can have multiple occurrences. For example, the first two rows in Table 3 both refer to the same resource event (Pete, normal, register, afternoon), indicating that Pete conducted an activity in the same execution context twice.

Table 3: A fragment of an example derived resource-event log

resource	case type	activity type	time type
...	...	...	...
Pete	normal	register	afternoon
Pete	normal	register	afternoon
Ann	normal	contact	afternoon
John	normal	check	morning
Sue	normal	check	morning
Bob	VIP	register	morning
John	normal	decide	morning
Sue	normal	decide	morning
Mary	VIP	check	afternoon
Mary	VIP	decide	afternoon
...	...	...	...

Organizational models can be discovered from an event log based on the similarities of resources characterized by a corresponding derived resource-event log. To do this, the following tasks need to be addressed:

1. *Determine execution contexts* by specifying the relevant case types, activity types, and time types based on the input event log;
2. *Discover resource groupings* by identifying clusters of resources who share similar behavior in process execution according to the derived resource-event log; and
3. *Associate resource groups with execution contexts* to describe the involvement of resource groups in process execution.

4.6. Evaluate Organizational Models

As discussed in the literature review, it remains an open issue how to evaluate discovered organizational models against input event logs. We address this gap by introducing two notions to organizational model mining, namely *fitness* and *precision* (cf. [5]), and their corresponding quantitative measures. The two notions are based on the new definition of organizational model and provide two perspectives for assessing an organizational model with respect to an event log.

Fitness. Fitness evaluates the completeness [31] of a model with respect to a log, i.e., to what degree behavior observed in the log is allowed by the model. To quantify fitness, we first introduce the notion of *conforming events* (Def. 8). Given a log and a model, an event in the log is conforming if its originating resource is allowed by the model to execute it. We define a measure for fitness

(Def. 9), which yields a value between 0 and 1. Note that only events with resource information (i.e., events in E_{res}) should be considered (hence fitness is only defined if there are events with resource information in the event log).

Definition 8 (Conforming Events). *Let $EL = (E, Att, \pi)$ be an event log and $OM = (RG, mem, cap)$ be an organizational model.*

$$E_{conf} = \left\{ e \in E_{res} \;\middle|\; \exists_{rg \in RG, co \in cap(rg)} [\pi_{res}(e) \in mem(rg) \land e \in [E]_{co}] \right\}$$

is the set of all conforming events. $E_{nconf} = E_{res} \setminus E_{conf}$ consists of all non-conforming events.

Definition 9 (Fitness). *Let $EL = (E, Att, \pi)$ be an event log with $E_{res} \neq \varnothing$. The fitness of an organizational model OM with respect to event log EL is*

$$fitness(EL, OM) = \frac{|E_{conf}|}{|E_{res}|}.$$

The fitness between a model and a log is good when most events in the log are conforming events. $fitness(EL, OM) = 1$ if resources only performed events in EL that they were allowed to perform according to OM. $fitness(EL, OM) = 0$ if no event in EL was executed by a resource actually allowed to perform it according to OM. Following the definitions, all events with resource information in the example event log (Table 2) are conforming events. Hence, the example organizational model shown in Figure 6 has a fitness of 1 with respect to the example event log.

Precision. Precision evaluates the exactness [31] of a model with respect to a log, i.e., the extent to which behavior allowed by the model is observed in the log. To quantify precision, we propose the notion of *candidate resources* (Def. 10). Given a log and a model, the candidate resources of an event refer to resources in the model who are allowed to perform the event. The idea is that a perfectly precise model allows exactly the behavior described in the log.

Definition 10 (Candidate Resources). *Let $EL = (E, Att, \pi)$ be an event log and $OM = (RG, mem, cap)$ be an organizational model. $cand \colon E \to \mathcal{P}(\mathcal{R})$ maps events onto sets of candidate resources (possibly empty). For each $e \in E$,*

$$cand(e) = \left\{ r \in \mathcal{R} \;\middle|\; \exists_{rg \in RG, co \in cap(rg)} [r \in mem(rg) \land e \in [E]_{co}] \right\}$$

is the set of candidate resources for event e. $cand(E) = \bigcup_{e \in E} cand(e)$ is the overall set of candidate resources.

We also introduce the notion of *allowed events* (Def. 11). Given a log and a model, an event in the log is an allowed event if it has at least one candidate resource in the model.

Definition 11 (Allowed Events). *Let $EL = (E, Att, \pi)$ be an event log and $OM = (RG, mem, cap)$ be an organizational model. Then $E_{allowed} = \{\, e \in E_{res} \mid cand(e) \neq \varnothing \,\}$ is the set containing all allowed events.*

Based on the above, the precision of a model with respect to an event log can be measured by considering the fraction of resources allowed by the model to perform events in the log (Def. 12). Like the fitness measure, the precision measure also yields a value between 0 and 1. Note that precision is only defined if there are allowed events in an event log according to a model.

Definition 12 (Precision). *Let $EL = (E, Att, \pi)$ be an event log and $OM = (RG, mem, cap)$ be an organizational model, with $E_{allowed} \neq \varnothing$. The precision of organizational model OM with respect to event log EL is*

$$precision(EL, OM) = \frac{1}{|E_{allowed}|} \sum_{e \in E_{conf}} \frac{|cand(E)| - |cand(e)| + 1}{|cand(E)|}.$$

Accordingly, $precision(EL, OM) = 1$ if and only if every allowed event in EL is a conforming event and each of them has no other candidate resource than the one who executed the event. On the other hand, $precision(EL, OM) = 0$ if and only if none of the allowed events is a conforming event. For instance, given the organizational model in Figure 6, the first event in the example log in Table 2 ("654423, register request, 29-08-2018 15:02, Pete, normal") has two candidate resources, Bob and Pete, and all events with resource information are allowed events. The precision of this model with respect to the log is 0.883, suggesting that the model allows some extra behavior to happen, in addition to that recorded in the event log.

For an organizational model discovered from an event log, fitness and precision can be used to assess its quality in terms of how it captures the information recorded in the log, i.e., the reality. A good discovered model is expected to describe the reality both completely (achieving high fitness) and exactly (achieving high precision). Fitness and precision can be incorporated into a single measure for an overall evaluation, e.g., by calculating the F1-score [31].

4.7. Analyzing Organizational Models

In this section, we discuss how organizational models can be analyzed to examine the behavior of resource groups. An organizational model outlines the groupings of resources and their capabilities in terms of process execution. We can extend a model by using event frequencies and temporal information about cases in an event log, and thus "replay" how resource groups in the model and their members participated in a process.

As a starting point, we introduce four quantitative measures that can be used for analyzing an organizational model. Note that all these measures only apply to events with resource information in an event log, i.e., events in E_{res}.

Group relative focus (Def. 13) specifies how much of the overall work by a resource group was performed in an execution context. It can be used to measure how a resource group distributed its workload across different execution contexts, i.e., work diversification of a group. Note that group relative focus is only defined if there are events executed by some member of the group.

Definition 13 (Group Relative Focus)**.** *Given event log $EL = (E, Att, \pi)$, execution contexts CO, and organizational model $OM = (RG, mem, cap)$, for any resource group $rg \in RG$ with $\{\, e \in E_{res} \mid \pi_{res}(e) \in mem(rg) \,\} \neq \varnothing$, its relative focus on execution context $co \in CO$ can be measured by*

$$RelFocus(rg, co) = \frac{|\{\, e \in [E_{res}]_{co} \mid \pi_{res}(e) \in mem(rg) \,\}|}{|\{\, e \in E_{res} \mid \pi_{res}(e) \in mem(rg) \,\}|}.$$

Group relative stake (Def. 14) specifies how much of work performed in an execution context was done by a resource group. It can be used to measure how the workload devoted to an execution context was distributed across different resource groups in an organizational model, i.e., work participation by the groups. Note that group relative stake is only defined if there are events having the execution context.

Definition 14 (Group Relative Stake)**.** *Given event log $EL = (E, Att, \pi)$, execution contexts CO, and organizational model $OM = (RG, mem, cap)$, for any resource group $rg \in RG$, its relative stake in execution context $co \in CO$, with $[E_{res}]_{co} \neq \varnothing$, can be measured by*

$$RelStake(rg, co) = \frac{|\{\, e \in [E_{res}]_{co} \mid \pi_{res}(e) \in mem(rg) \,\}|}{|[E_{res}]_{co}|}.$$

Group coverage (Def. 15) specifies the proportion of members of a resource group that performed in an execution context. Note that group coverage is only defined if there are resources in the resource group.

Definition 15 (Group Coverage)**.** *Given event log $EL = (E, Att, \pi)$, execution contexts CO, and organizational model $OM = (RG, mem, cap)$, for any resource group $rg \in RG$ with $mem(rg) \neq \varnothing$,* *the proportion of group members covered by execution context $co \in CO$ can be measured by*

$$Cov(rg, co) = \frac{\left|\{\, r \in mem(rg) \mid \exists_{e \in [E_{res}]_{co}} \pi_{res}(e) = r \,\}\right|}{|mem(rg)|}.$$

Group member contribution (Def. 16) specifies how much work conducted in an execution context by a group was performed by a specific group member. It can be used to measure how a group's workload related to an execution context was distributed across its members. Note that group member contribution is only defined if there are events executed by some member of the resource group in the execution context.

Definition 16 (Group Member Contribution)**.** *Given event log $EL = (E, Att, \pi)$, execution contexts CO, and organizational model $OM = (RG, mem, cap)$, for resource group $rg \in RG$ and execution context $co \in CO$, with $\{\, e \in [E_{res}]_{co} \mid \pi_{res}(e) \in mem(rg) \,\} \neq \varnothing$, the contribution of a group member $r \in mem(rg)$ can be measured by*

$$MemContr(r, rg, co) = \frac{|\{\, e \in [E_{res}]_{co} \mid \pi_{res}(e) = r \,\}|}{|\{\, e \in [E_{res}]_{co} \mid \pi_{res}(e) \in mem(rg) \,\}|}.$$

Consider the example organizational model in Figure 6. For resource group "Group 0" and one of its capabilities (VIP, register, morning), we have:

- $RelFocus$("Group 0", (VIP, register, morning)) = 0.333,
- $RelStake$("Group 0", (VIP, register, morning)) = 1.0,
- Cov("Group 0", (VIP, register, morning)) = 0.5, and
- $MemContr$(Bob, "Group 0", (VIP, register, morning)) = 0,
- $MemContr$(Pete, "Group 0", (VIP, register, morning)) = 1.0.

As shown above, resources in "Group 0" devoted 33% of their total workload (indicated by *RelFocus*) to carrying out activities related to "registering requests for VIP cases in the morning"; "Group 0" is the only group that contributed to such work (indicated by *RelStake*) in the process; only 50% of the group members (indicated by *Cov*) actually participated in this type of work, and that is resource Pete (indicated by *MemContr*). Furthermore, these model analysis measures can also be used to "diagnose" the differences between an organizational model and a log. In the example organizational model, we can find that "Group 0" is the only one that has a comparatively low group coverage in terms of its capabilities — the model considers *both* of its members, Bob and Pete, capable of performing in *both* execution contexts, but the event log does not show such behavior. This explains the imperfect precision (0.883). Also, if the example model is a discovered model, then the revealed differences can inform how to improve the discovery method.

5. Approach

5.1. Design of Approach

Figure 7 shows an overview of our approach to realize the proposed *OrdinoR* framework. First, an event log with the standard attributes (*case*, *act*, *time*) and resource information (*res*) is used as input for learning a set of execution contexts. A resource-event log can then be derived and utilized for discovering resource groups. Next, the discovered resource groups are "profiled" using execution contexts to describe the group capabilities in process execution. Finally, an organizational model is constructed and subsequently evaluated and analyzed by applying the measures in the framework.

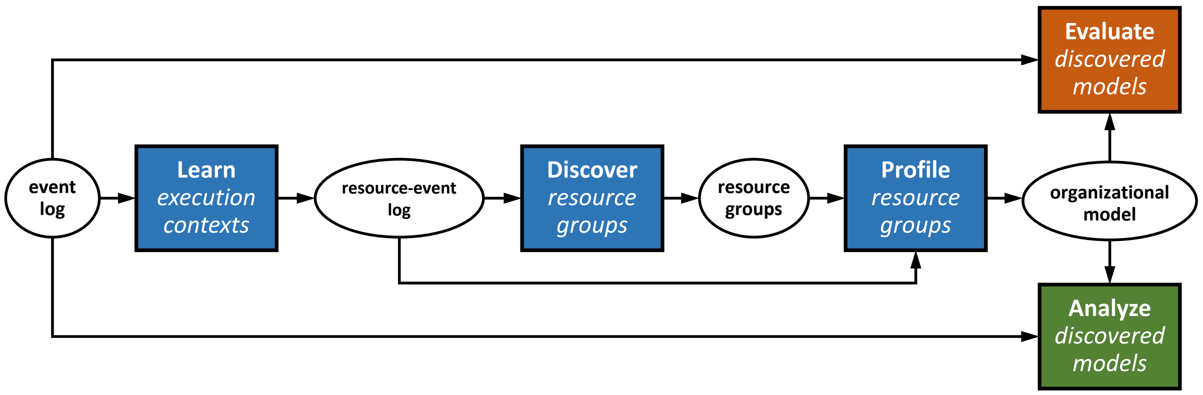


Figure 7: An overview of how to use the *OrdinoR* framework

Learning execution contexts. Execution contexts can be determined directly if prior knowledge about the categorization of cases, activities, and time periods is available. For instance, managers can decide time types based on the seasonal patterns of how employees work.

When such knowledge is limited, there are process mining techniques that can be applied to automatically learn a meaningful collection of case types, activity types, and time types. Examples include mining local process models [32] to find subsets of process activities representing frequent patterns (informing activity types), and using trace clustering [33] to find coherent sets of cases (informing case types). However, deriving a general method for learning execution contexts is a far from trivial problem. For validation purposes, we discuss three approaches.

1. ATonly considers only activity types and defines each activity label as an activity type. The other two dimensions are omitted, i.e., all events will have the special type $\perp$ as the case type and time type (see Def. 3). ATonly can be applied as a basic solution, since activity information is standard for event logs.
2. CT+AT+TT (case attribute) defines each activity label as an activity type and specifies time types according to the seven week days, i.e., time types will be "Monday", "Tuesday", etc. Case types are defined based on a selected case attribute that is meaningful for distinguishing between different variants of cases in an input event log.
3. CT+AT+TT (trace clustering) is similar to the foregoing approach, except that case types are defined by applying context-aware trace clustering [33]. Trace clustering assigns a cluster label to each case and this label is considered the case type for all events in the case.

Discovering resource groups. Resource groups are discovered by identifying resources with similar behavior according to a resource-event log derived from the previous step. To this end, we first build a set of features from a derived resource-event log to characterize individual resource behavior. We consider the variety of execution contexts in which a resource worked and the occurrences of resource events. As a result, we can construct a feature matrix [31] where each row corresponds to a resource, each column corresponds to an execution context, and each entry value denotes the number of occurrences of the resource event. Table 4 shows an example feature matrix characterizing the behavior of six resources.

With a resource feature matrix, we can address the task of identifying similar resources by applying established cluster analysis techniques in data mining [31]. In our validation, we adopt Agglomerative Hierarchical Clustering (AHC) [31] and Model-based Overlapping Clustering (MOC) [34]. AHC results in disjoint resource groups and MOC results in potentially overlapping groups.

Profiling resource groups. The final step is to profile each discovered resource group with a set of relevant execution contexts characterizing the group's capabilities in process execution.

We first consider a strategy, namely FullRecall, which accounts for all historical behavior by any member of a resource group. Given a derived resource-event log $RL(EL, CO)$, the set of execution

Table 4: An example resource feature matrix related to the example resource-event log in Table 3. Note that a resource feature matrix in practice will usually have more rows and columns

resource	(normal, register, afternoon)	(normal, contact, afternoon)	(normal, check, morning)	(VIP, register, morning)	(normal, decide, morning)	(VIP, check, afternoon)	(VIP, decide afternoon)
Ann	0	1	0	0	0	0	0
Bob	0	0	0	1	0	0	0
John	0	0	1	0	1	0	0
Mary	0	0	0	0	0	1	1
Pete	2	0	0	0	0	0	0
Sue	0	0	1	0	1	0	0

contexts for profiling a group *rg* is

$$cap(rg) = \left\{ co \in CO \;\middle|\; \exists_{r \in mem(rg)} (r, co) \in RL(EL, CO) \right\}.$$

Applying this definition, a resulting organizational model will capture all observed behavior recorded in the log and will thus achieve the best fitness. However, the use of FullRecall risks linking a resource group with an excessive number of execution contexts. This is because FullRecall considers every resource event related to any group member, even if that event may represent rare behavior.

Hence, we introduce another strategy OverallScore that ranks execution contexts according to *how frequent* and *how popular* they are with respect to the members of a resource group. If the process activities within an execution context were mostly taken by a specific group, or by the majority of members in a group, then this execution context is likely associated with the group. OverallScore can be formalized as selecting execution contexts based on the weighted average of two model analysis measures, *group relative stake* and *group coverage*, i.e.,

$$cap(rg) = \{ co \in CO \mid \omega_1 \cdot RelStake(rg, co) + \omega_2 \cdot Cov(rg, co) \geq \lambda \},$$

where λ is a threshold in range $(0, 1)$, and ω_1, ω_2 are non-negative weights satisfying $\omega_1 + \omega_2 = 1$.

Applying OverallScore links a resource group to only its most relevant execution contexts and hence leads to models with relatively balanced fitness and precision.

Table 5 presents an example of applying these two strategies to profile a group with two resources, John and Sue.

Table 5: An example of profiling a resource group with two members, applying both FullRecall and OverallScore ($\omega_1 = \omega_2 = 0.5, \lambda = 0.8$)

$mem(rg)$	$cap(rg)$ (applying FullRecall)	$cap(rg)$ (applying OverallScore)
"John", "Sue"	(normal, check insurance, Thursday) (normal, accept claim, Thursday) (normal, reject claim, Thursday)	(normal, check insurance, Thursday)

5.2. Implementation

We developed an open-source software tool implementing the approach above. It consists of (1) an extensible Python library[2] and (2) a web-based application[3], enabling users to perform organizational model mining tasks and visualize the outcomes. The tool was built following a modular design and therefore can be extended to integrate new methods and measures for organizational model mining in the future.

6. Experiments

We conducted extensive experiments with the aim to demonstrate how organizational models can be discovered from event log data by applying our approach, and how those models can be evaluated and analyzed using the proposed measures in the framework.

6.1. Experiment Dataset

Two real-life event logs were used for experiments, which are both publicly available online. The first log (WABO[4]) records the receipt phase of a building permit process in an anonymous Dutch municipality. The second log (BPIC17[5]) records data related to a loan application process in a Dutch financial institute. Both of them satisfy the basic requirements for event logs defined previously, containing case identifiers, activity labels, timestamps, and resource information. In addition, they also carry several case-level attributes, such as the one recording loan purposes in BPIC17.

[2]The OrdinoR library: `https://royjy.me/to/ordinor`

[3]Web-based tool for organizational model mining: `https://royjy.me/to/arya`

[4]'WABO', CoSeLoG project: `https://doi.org/10.4121/uuid:a07386a5-7be3-4367-9535-70bc9e77dbe6`

[5]BPI Challenge 2017: `https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b`

To obtain the experiment dataset, we preprocessed the original logs to filter out redundant event data, keeping only events recording the completion of activities. This filtering step ensures that each activity instance in process execution is counted exactly once. Table 6 reports the basic statistics of the preprocessed event log dataset.

Table 6: Basic statistics of the preprocessed event logs for experiments

Log	#cases	#events	#activities	#resources
WABO	1,434	8,577	27	48
BPIC17	31,509	475,306	24	144

6.2. Experiment Setup

The approach designed for validation (Section 5.1) includes several alternative methods at each intermediate step of model discovery. In the experiments, we tested all combinations of these alternatives. Figure 8 depicts an overview of the experiment setup — each organizational model is discovered using a selected execution contexts learning method, a resource groups discovery method, and a resource groups profiling method, respectively. Discovered models are then evaluated and analyzed.

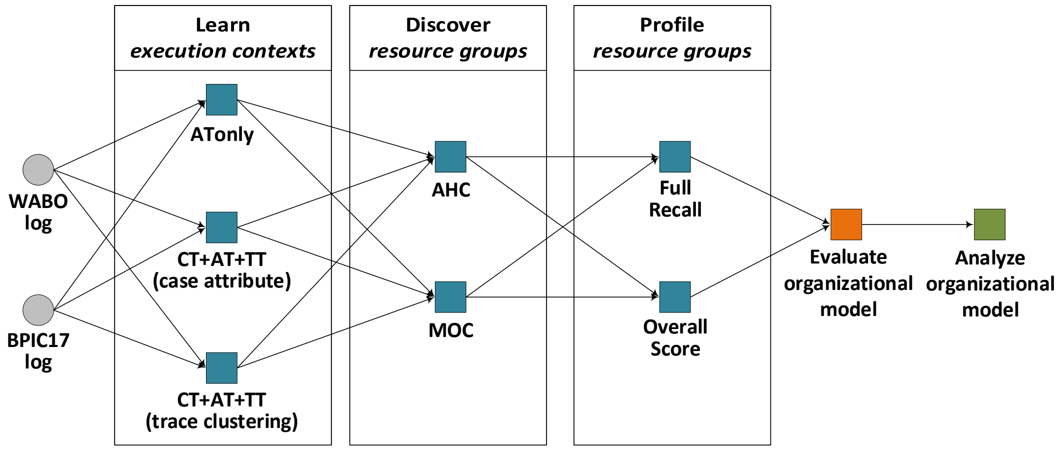


Figure 8: An overview of the experiment setup: each path in the graph specifies a selection of methods applied on an input event log

Combining all these alternatives resulted in a considerable number of organizational models to be discovered, evaluated, and analyzed. Therefore, we applied the principles of scientific workflow to efficiently conduct the experiments. The use of scientific workflow also benefits future research that wishes to replicate or modify our experiments. The following shows how we configured the methods at each intermediate step in model discovery[6].

In terms of methods for learning execution contexts, ATonly requires no specific configuration. For CT+ AT+TT (case attribute), we selected the attribute recording the channel of environmental permit application for log WABO, and the attribute recording the loan purpose of applicants for log BPIC17. For CT+AT+TT (trace clustering), we applied the original configurations in the paper of Bose and Van der Aalst [33].

In terms of discovering resource groups, we applied the same configuration for both clustering techniques, AHC and MOC. The Euclidean distance was selected as the proximity measure, and the number of resource groups (clusters) was decided by using cross-validation [31], in which the potential group number was tested between 2 and 10.

In terms of profiling resource groups, FullRecall requires no specific configuration. For OverallScore, we performed a grid search within the range $[0.1, 0.9]$ using a search step of 0.1 to determine the weights (ω_1 and ω_2) and the threshold (λ).

6.3. Results and Findings

6.3.1. Model Evaluation and Comparison

In the experiments, we discovered and evaluated a total of 24 organizational models (12 per log). We compared models discovered from the same event log to investigate how different model discovery methods impact model quality. The baseline models used in the comparisons were the ones with the highest F1-scores (the best quality), as shown in Table 7.

Table 7: Discovered models with the best quality used as baselines in the comparisons

Log	Configuration			Model statistics		Evaluation		
				#execution contexts	#groups	fitness	precision	F1
WABO	CAT (tc)	MOC	OS	307	9	0.876	0.577	0.696
BPIC17	CAT (tc)	AHC	OS	1884	10	0.831	0.641	0.724

Configuration: CAT = CT+AT+TT, tc = trace clustering; OS = OverallScore

We compared between three sets of discovered organizational models with respect to the three steps in model discovery. Note that models in the same set differ only in terms of the method

[6]Details of experiment configuration can be found here: `https://royjy.me/to/om2-experiments`

Table 8: Comparing models discovered from applying ATonly, CT+AT+TT (case attribute), or CT+AT+TT (trace clustering) for learning execution contexts

Log	Configuration			Model statistics		Evaluation		
				#execution contexts	#groups	fitness	precision	F1
WABO	ATonly	MOC	OS	27	9	0.947	0.407	0.569
	CAT (ca)			263	9	0.843	0.448	0.586
	<u>CAT (tc)</u>			307	9	<u>0.876</u>	<u>0.577</u>	<u>0.696</u>
BPIC17	ATonly	AHC	OS	24	10	0.923	0.530	0.673
	CAT (ca)			2020	10	0.810	0.629	0.708
	<u>CAT (tc)</u>			1884	10	<u>0.831</u>	<u>0.641</u>	<u>0.724</u>

Configuration: CAT = CT+AT+TT, ca = case attribute, tc = trace clustering; OS = OverallScore

applied for that particular step. In the following tables, results of the baseline models are underlined to aid comparisons.

Learning execution contexts. (Table 8) Models produced from applying ATonly have a higher fitness compared to the baseline. This is because applying ATonly led to a relatively coarser collection of execution contexts, thus related the same number of events to fewer execution contexts. Consequently, more events can be fit by a model. However, model precision was harmed due to having excessive candidate resources for events.

On the other hand, comparing between models generated from using CT+ AT+TT (case attribute) and CT+AT+TT (trace clustering), we can observe that the latter ones have comparatively higher values in both fitness and precision. This indicates using a trace clustering technique can lead to more reasonable case types, which better capture the categorization of different cases in comparison to relying on a single case attribute.

Discovering resource groups. (Table 9) The baseline models produced from applying MOC have higher fitness but lower precision, compared to those from AHC. This is expected, because resource groups discovered using MOC can be overlapped, whereas the AHC ones are disjoint. When groups are overlapped, a resource can be a member of more than one group, and can thus be linked with more execution contexts. As a result, models with overlapping groups can potentially fit more events (increasing fitness) but link excessive candidate resources to events (reducing precision).

Profiling resource groups. (Table 10) Using FullRecall resulted in models with perfect fitness, but this method sacrifices much precision, because resource groups are usually linked with a large number of irrelevant execution contexts. This is similar to the concept of "flower models" [5]

Table 9: Comparing models discovered from applying AHC or MOC for discovering resource groups

Log	Configuration			Model statistics		Evaluation		
				#execution contexts	#groups	fitness	precision	F1
WABO	CAT (tc)	AHC	OS	307	9	0.815	0.595	0.687
		<u>MOC</u>		307	9	<u>0.876</u>	<u>0.577</u>	<u>0.696</u>
BPIC17	CAT (tc)	<u>AHC</u>	OS	1884	10	<u>0.831</u>	<u>0.641</u>	<u>0.724</u>
		MOC		1884	8	0.957	0.406	0.571

Configuration: CAT = CT+AT+TT, tc = trace clustering; OS = OverallScore

Table 10: Comparing models discovered from applying FullRecall or OverallScore for profiling resource groups

Log	Configuration			Model statistics		Evaluation		
				#execution contexts	#groups	fitness	precision	F1
WABO	CAT (tc)	MOC	FR	307	9	1.000	0.067	0.125
			<u>OS</u>	307	9	<u>0.876</u>	<u>0.577</u>	<u>0.696</u>
BPIC17	CAT (tc)	AHC	FR	1884	10	1.000	0.169	0.290
			<u>OS</u>	1884	10	<u>0.831</u>	<u>0.641</u>	<u>0.724</u>

Configuration: CAT = CT+AT+TT, tc = trace clustering; FR = FullRecall, OS = OverallScore

in process model discovery. On the other hand, the baseline models (applying OverallScore) have better precision while maintaining moderate fitness, since execution contexts were selectively linked to resource groups based on frequency and popularity.

Conclusion. Fitness and precision proposed in our framework can provide a generic and consistent basis for evaluating organizational models discovered using different mining algorithms, which fulfills the vital missing part of model evaluation in the state-of-the-art.

6.3.2. Model Diagnosis

In the experiments for model analysis, we selected a discovered model with the lowest F1-score (the worst quality overall) and used the model analysis measures to identify the possible reasons. This model was derived from log WABO applying CT+AT+TT (case attribute)-MOC-FullRecall. It consists of 48 resources in 9 groups with overlaps and has fitness = 1.0, precision = 0.036, and F1-score = 0.069. The perfect fitness and poor precision of this model imply that some resource groups and execution contexts were inappropriately linked during discovery, causing certain events in the log to have excessive candidate resources (Def. 10). To identify such groups and execution contexts, we applied the *group relative stake* (Def. 14) and *group coverage* (Def. 15) measures.

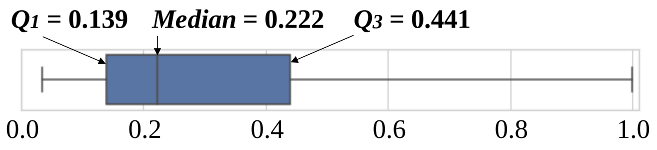
Group relative stake can reveal whether a group had only little contribution to an execution context, while group coverage can reveal whether only a small number of group members were involved in an execution context.

Table 11 presents the average values of the two measures for each group in the model, along with the rankings of groups based on those values. We can tell that "Group 3" is the one with the comparatively *lowest* group relative stake and group coverage among all groups.

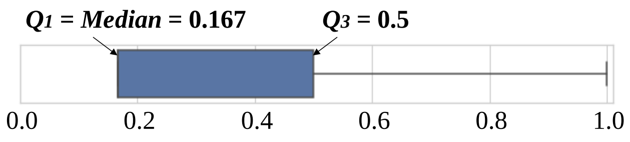
Table 11: Average group relative stake and group coverage of the resource groups in the selected model

resource group	#members	#execution contexts	average group relative stake	rank of avg. group relative stake	average group coverage	rank of avg. group coverage
Group 0	1	68	0.220	9	1.000	1
Group 1	3	133	0.280	7	0.569	3
Group 2	37	234	0.684	1	0.124	9
Group 3	**6**	**128**	**0.306**	**6**	**0.359**	**6**
Group 4	5	151	0.355	4	0.440	4
Group 5	2	122	0.236	8	0.693	2
Group 6	5	150	0.350	5	0.405	5
Group 7	12	209	0.476	2	0.285	8
Group 8	8	203	0.407	3	0.330	7

We therefore focused on "Group 3" by investigating all its associated execution contexts. The box plots in Figure 9 summarize the distributions of group relative stake and group coverage values, respectively. Figure 9a shows that, in half of its associated 128 execution contexts, "Group 3" contributed a fair amount of work (more than 22.2%). However, Figure 9b indicates that most execution contexts involved no more than half of the group members ($Q_3 = 0.5$). More specifically, the median value (0.167, or 1/6) indicates that the work in these execution contexts was performed by only one group member.



(a) group relative stake



(b) group coverage

Figure 9: Distributions of group relative stake and group coverage of "Group 3"

We concluded from the results that the selected low-quality model over-generalized the capability of "Group 3", linking the group with execution contexts that were only specific to a small number of group members. We furthered our analysis by examining individual execution contexts and individual group members of “Group 3”. Table 12 shows the bottom-five execution contexts with the lowest group relative stake and group coverage. The group relative stake values indicate that less than 5% of the total work performed in these execution contexts was actually committed by resources from “Group 3”. Moreover, for each of the execution contexts, only 1 of 6 group members performed the work, as indicated by group coverage.

Table 12: Execution contexts linked to “Group 3” with the lowest group relative stake and group coverage

execution context	group relative stake	group coverage	member contribution
(CT.Desk, AT.Confirmation of receipt-complete, TT.4)	0.034	0.167	Resource 21 (100%)
(CT.Desk, AT.T02 Check confirmation of receipt-complete, TT.4)	0.037	0.167	Resource 19 (100%)
(CT.Desk, AT.Confirmation of receipt-complete, TT.2)	0.040	0.167	Resource 18 (100%)
(CT.Desk, AT.T06 Determine necessity of stop advice-complete, TT.1)	0.045	0.167	Resource 18 (100%)
(CT.Desk, AT.T02 Check confirmation of receipt-complete, TT.3)	0.048	0.167	Resource 19 (100%)

Conclusion. The above analyses can be replicated on other resource groups and their associated execution contexts in the selected model. We can therefore extend our previous model evaluation through explaining the discrepancies between the model and the actual behavior in the log.

7. Discussion

7.1. Threats to Validity

The first concern is internal validity. Event logs — however extensive and detailed — can only capture certain information about actual business processes, the participating employees, or their organizations. Hence, confounding factors may exist in terms of how employees are organized into groups and involved in process execution. But these factors are not reflected by organizational models discovered from event logs. Clearly, without additional inputs, knowledge discovered from a set of data can hardly transcend the scope of the original data collection. Therefore, to mitigate the impact of confounding factors, it is vital to understand the scope and characteristics of event logs and their corresponding processes when applying our approach and interpreting the outputs.

Another concern is external validity. While the selected techniques and data are suitable examples of the respective artifacts in our research, experimental results are prone to few circumstances and therefore may not be generalized easily. Also, the results may risk being biased due to the specificity of the selected techniques and data. To address these limitations, it is essential to first explore the configurability of the *OrdinoR* framework. We used rigorous formalization to define the requirements for event log data used in the framework. We have also outlined the key tasks of discovering organizational models, which provide initial criteria for selecting or developing the corresponding techniques. Besides, in the experiments we have shown how data mining techniques, e.g., cross-validation and grid search, can be useful for configuring specific parameters in the proposed approach. Based on the above, an extensive benchmark can be performed in future work using more event logs from different contexts and applying a wider variety of techniques to study the impact of various configurations. Also, artificial data from process simulation can be included as a complement to real-life data to cover unusual scenarios. Still, proper configuration of the framework and the approach requires understanding the context of input data, and may need to be conducted case by case. Finally, it is worth noting that experimentation is inevitably limited when investigating how to apply the framework to real problems — future evaluations need to involve case studies in real-world organizations.

7.2. Future Work

In addition to those immediate next steps outlined above, there are many possibilities for future research opened up by the *OrdinoR* framework. Among others, we highlight the *conformance checking of organizational models*, i.e., comparing the modeled-behavior with the real-behavior as reflected in event logs [5]. To begin with, organizational models are constructed from existing employee groupings, such as departments, business roles, and project teams. This can be done by identifying (1) employee groups and their members involved in process execution, and (2) the patterns or rules regarding how process activities are performed based on the grouping. Data other than event logs may be needed, e.g., documents about work distribution rules and timetables showing employee shifts.

Then, given an organizational model, either discovered or based on existing employee groups, conformance checking can be performed. More specifically, fitness and precision in the *OrdinoR* framework can be used for *global conformance checking* — measuring the extent of commonalities between the modeled and actual behavior of human resource groups. The model analysis measures, e.g., group coverage, can be used for *local conformance checking* — showing where and how the modeled human resource groups differ from reality.

Hence, combining conformance checking with organizational model mining capability, the *OrdinoR* framework lays the foundation for systematically exploiting process execution data to guide the design of organizational structures (covering, e.g., role designation and employee team composition) and staff deployment alongside changing business processes. Organizations can thus be empowered to evolve organizational structures toward process improvement, and to iteratively evaluate their decisions to enhance coordination, team effectiveness, and work satisfaction. To support this capacity-building, future work can investigate additional information needed in event logs to measure those goals.

8. Conclusion

We have proposed a framework based on a new definition of organizational models in the context of business processes. Compared to existing research, our organizational models establish a linkage from human resources via resource groups to activities, case types, and time periods, and can thus describe the involvement of resource groups in process execution. Another aspect of novelty concerns the model evaluation and analysis measures for organizational models. These measures contribute a rigorous and generic means for assessing the quality of discovered organizational models, and support for understanding the actual behavior of groups and their members in process execution. Furthermore, our framework opens the door to conformance checking of organizational models and can therefore bridge process analytics with analyses on organizational structures.

Acknowledgments

The reported research is part of a PhD project supported by an Australian Government Research Training Program (RTP) Scholarship. We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research. This work is also supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61972427 and U20A6003, and by the Research Foundation of Science and Technology Plan Project in Guangdong Province under Grant No. 2020A0505100030.

References

[1] M. Dumas, M. L. Rosa, J. Mendling, H. A. Reijers, Fundamentals of Business Process Management, 2nd Edition, Springer Publishing Company, Incorporated, 2018.

[2] R. L. Daft, J. Murphy, H. Willmott, Organization theory and design, Cengage learning, 2010.

[3] T. H. Davenport, J. Harris, J. Shapiro, Competing on talent analytics, Harvard Business Review 88 (10) (2010) 52–58.

[4] A. Pika, M. Leyer, M. T. Wynn, C. J. Fidge, A. H. M. ter Hofstede, W. M. P. van der Aalst, Mining resource profiles from event logs, ACM Trans. Manage. Inf. Syst. 8 (1) (2017) 1:1–1:30.

[5] W. M. P. van der Aalst, Process Mining: Data Science in Action, Springer, Berlin, 2016.

[6] M. Song, W. M. P. van der Aalst, Towards comprehensive support for organizational mining, Decis. Support Syst. 46 (1) (2008) 300–317.

[7] S. Schönig, C. Cabanillas, S. Jablonski, J. Mendling, A framework for efficiently mining the organisational perspective of business processes, Decis. Support Syst. 89 (2016) 87–97.

[8] A. Appice, Towards mining the organizational structure of a dynamic event scenario, Journal of Intelligent Information Systems 50 (1) (2018) 165–193.

[9] J. Yang, C. Ouyang, M. Pan, Y. Yu, A. H. M. ter Hofstede, Finding the "liberos": Discover organizational models with overlaps, in: International Conference on Business Process Management (BPM 2018), Springer, 2018, pp. 339–355.

[10] A. Burattin, A. Sperduti, M. Veluscek, Business models enhancement through discovery of roles, in: IEEE Symposium on Computational Intelligence and Data Mining (CIDM), 2013, pp. 103–110.

[11] W. M. P. van der Aalst, H. A. Reijers, M. Song, Discovering social networks from event logs, Computer Supported Cooperative Work 14 (6) (2005) 549–593.

[12] D. R. Ferreira, C. Alves, Discovering user communities in large event logs, in: International Conference on Business Process Management, 2011, pp. 123–134.

[13] R. Liu, S. Agarwal, R. R. Sindhgatta, J. Lee, Accelerating collaboration in task assignment using a socially enhanced resource model, in: International Conference on Business Process Management (BPM 2013), Springer, 2013, pp. 251–258.

[14] L. T. Ly, S. Rinderle, P. Dadam, M. Reichert, Mining Staff Assignment Rules from Event-Based Data, in: International Conference on Business Process Management Workshops (BPM 2005), Springer, 2005, pp. 177–190.

[15] Z. Huang, X. Lu, H. Duan, Mining association rules to support resource allocation in business process management, Expert Syst. Appl. 38 (8) (2011) 9483–9490.

[16] J. Nakatumba, W. M. P. van der Aalst, Analyzing resource behavior using process mining, in: International Conference on Business Process Management, Springer, 2009, pp. 69–80.

[17] S. Suriadi, M. T. Wynn, J. Xu, W. M. P. van der Aalst, A. H. M. ter Hofstede, Discovering work prioritisation patterns from event logs, Decis. Support Syst. 100 (2017) 77–92.

[18] T. Jin, J. Wang, L. Wen, Organizational modeling from event logs, in: International Conference on Grid and Cooperative Computing (GCC 2007), 2007, pp. 670–675.

[19] J. Ye, Z. Li, K. Yi, A. Abdulrahman, Mining resource community and resource role network from event logs, IEEE Access 6 (2018) 77685–77694.

[20] A. Baumgrass, Deriving Current State RBAC Models from Event Logs, in: International Conference on Availability, Reliability and Security (ARES 2011), 2011, pp. 667–672.

[21] W. Zhao, Q. Lin, Y. Shi, X. Fang, Mining the Role-Oriented Process Models Based on Genetic Algorithm, in: International Conference on Advances in Swarm Intelligence (ICSI 2012), Springer, 2012, pp. 398–405.

[22] Z. Ni, S. Wang, H. Li, Mining organizational structure from workflow logs, in: International Conference on e-Education, Entertainment and e-Management (ICeEEM), 2011, pp. 222–225.

[23] M. Li, L. Liu, L. Yin, Y. Zhu, A process mining based approach to knowledge maintenance, Information Systems Frontiers 13 (3) (2011) 371–380.

[24] C. Hanachi, W. Gaaloul, R. Mondi, Performative-Based Mining of Workflow Organizational Structures, in: International Conference on E-Commerce and Web Technologies (EC-Web 2012), Springer, 2012, pp. 63–75.

[25] R. Sellami, W. Gaaloul, S. Moalla, An Ontology for Workflow Organizational Model Mining, in: IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2012), 2012, pp. 199–204.

[26] L. Bouzguenda, M. Abdelkafi, An agent-based approach for organizational structures and interaction protocols mining in workflow, Soc. Netw. Anal. Min. 5 (1) (2015) 10:1–10:22.

[27] K. Peffers, T. Tuunanen, M. A. Rothenberger, S. Chatterjee, A design science research methodology for information systems research, J. Manag. Inf. Syst. 24 (3) (2007) 45–77.

[28] A. R. Hevner, S. T. March, J. Park, S. Ram, Design science in information systems research, MIS Quarterly: Management Information Systems 28 (1) (2004) 75–105.

[29] W. Zhao, X. Zhao, Process mining from the organizational perspective, in: Foundations of Intelligent Systems, Springer, 2014, pp. 701–708.

[30] W. M. P. van der Aalst, Process cubes: Slicing, dicing, rolling up and drilling down event data for process mining, in: M. Song, M. T. Wynn, J. Liu (Eds.), Asia Pacific Business Process Management, Springer International Publishing, Cham, 2013, pp. 1–22.

[31] J. Han, J. Pei, M. Kamber, Data mining: concepts and techniques, Elsevier, 2011.

[32] N. Tax, N. Sidorova, R. Haakma, W. M. P. van der Aalst, Mining local process models, J. Innovation Digital Ecosyst. 3 (2) (2016) 183–196.

[33] R. P. J. C. Bose, W. M. P. van der Aalst, Context Aware Trace Clustering: Towards Improving Process Mining Results, in: SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, 2009, pp. 401–412.

[34] A. Banerjee, C. Krumpelman, J. Ghosh, S. Basu, R. J. Mooney, Model-based overlapping clustering, in: ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 2005, pp. 532–537.

View publication stats