Business Models Enhancement through Discovery of Roles

Andrea Burattin
Student Member, IEEE
Department of Mathematics
University of Padua, Italy

Alessandro Sperduti
Senior Member, IEEE
Department of Mathematics
University of Padua, Italy

Marco Veluscek
Research and Development Department
SIAV S.p.A.
Rubano, Italy

***Abstract*—Control flow discovery algorithms are able to reconstruct the workflow of a business process from a log of performed activities. These algorithms, however, do not pay attention to the reconstruction of roles, i.e. they do not group activities according to the skills required to perform them. Information about roles in business processes is commonly considered important and explicitly integrated into the process representation, e.g. as swimlanes in BPMN diagrams. This work proposes an approach to enhance a business process model with information on roles. Specifically, the identification of roles is based on the detection of *handover of roles*. On the basis of candidates for roles handover, the set of activities is first partitioned and then subsets of activities which are performed by the same originators are merged, so to obtain roles. All significant partitions of activities are automatically generated. Experimental results on several logs show that the set of generated roles is not too large and it always contains the correct definition of roles. We also propose an entropy based measure to rank the candidate roles which returns promising experimental results.**

***Index Terms*—process mining, process enhancement, organizational mining, social network analysis.**

I. Introduction and Problem Description

Process mining is a relatively young field, bridging business process modeling and data mining. A growing community of both academic and industrial partners, coming from different fields, is actively working on several research topics [1].

Process mining, actually, is a very broad term and involves several aspects and techniques. Specifically, it is possible to classify process mining in three main activities:

1) *discovery* which aims, given a log as input, at building a representation of the underlying business process model;
2) *conformance* techniques are useful to compare a reference process model with respect to a given log (i.e., the ideal process against what happens in reality);
3) *enhancement* which enrich a process model, given as input, with new information extracted starting from an input log.

Apart from this characterization, it is interesting to note that several *perspectives* might be involved in process mining. In particular, it is possible to concentrate on:

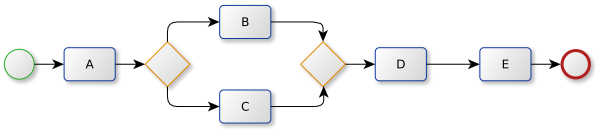
- the *control-flow*, which is a (possibly graphical) representation of the business process model (i.e., the ordering of activities);
- the *organizational perspective*, which focuses on the interactions among activities originators;
- focusing on *cases* (single process instances) may help identifying peculiarities based on specific characteristics (for example, which case conditions lead to a particular path of the process model);
- the *time perspective* is extremely useful to measure and monitor the process, for example to find bottlenecks or predict the remaining time of a case.

In this paper, we concentrate on the *enhancement* of a process from the *organizational perspective*. Specifically, we present an approach which, given a process model and a log in input, tries to partition the set of activities of the process in "swimlanes". This partitioning is performed by grouping originators in roles and associating activities with the corresponding role.

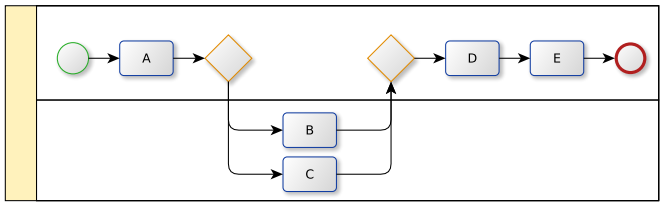
The partitioning proposed in this work is based on the identification of roles and this is, in turn, based on the observation of the distribution of originators over activities and roles. This division is extremely important and gives new detailed insights on the process model (which can be extracted using discovery techniques). For example, it is possible to compare the actual roles distribution with the mined ones or to analyze the proposed roles in order to improve the current organization.

The approach proposed in this work, summarized in Figure 1, is composed of two phases: it starts from the original process model and, in the first phase, each edge of the process model is weighted according to the corresponding level of handover of role. Edges with weight below a threshold are removed from the model. Resulting connected components are considered as belonging to the same role. The second phase of the approach aims at merging components that, in the original process model, were not close each other.

The remainder of this paper is structured as follows: Section II presents some works related to social network analysis and organizational mining. Section III introduces the general framework and definitions that will be used throughout this paper and Section IV describes the general approach and the metrics employed. The paper continues in Section V with the description of the proposed algorithms. Experimental results are described in Section VI, and conclusions are presented in Section VII.



(a) Original process model.



(b) Expected result, with activities partitioned in roles.

Fig. 1. Input and expected output of the approach presented in this paper.

II. Related Work

The *organizational perspective* of process mining aims at discovery relations among activity originators. Typically, these activities involve several approaches, such as classification of users in roles and Social Network Analysis [2].

In [3], Song and van der Aalst present an exhaustive characterization of organizational mining approaches. In particular, three different types of approaches are presented: *a*) organizational model mining; *b*) social network analysis; and *c*) information flows between organizational entities.

Organizational model mining consists in grouping users with similar characteristics. This grouping can rely on the similarity of activities performed (task based) or on working together on the same process instance (case based). The basic idea of social network analysis [4] is to discover how the work is handled between different originators. Several metrics are employed to point out different perspectives of the social network. Examples of such metrics are the *handover of work* (when the work is started by one user and is completed by another), *subcontracting* (when activities are performed by user a, then user b and then a again) and *working together* (users involved in the same case). The information collected in social networks can be aggregated in order to produce *organizational entities* (such as roles or organizational unit). These entities are useful to provide insights at a higher abstraction level. Organizational entities are constructed considering a metric and the deriving social network and aggregating nodes. These new connections can be weighted according to the weights of the originating network.

None of the above papers specifically addresses the problem of discovering roles in business processes.

III. Working Framework

Given a business process P, it is possible to identify its set of tasks (or activities) A and the set U with all the involved originators (e.g. person, resources, ...). In this context, the complete set of observable events, generated by P, is defined as $E = A \times U$.

A process can generate a log $L = \{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$, which is defined as a set of traces. Each element of the log identifies a case (i.e. a process instance) of the observed model. A trace $\mathbf{e} = \langle e_1, \ldots, e_m \rangle$ is a sequence of events, where $e_j \in E$ represents the jth event of the sequence. With $e_i \in \mathbf{e}$ we indicate that event e_i is contained in the sequence $\mathbf{e}$.

Given a process model P, let $\mathcal{D}(P)$ be the set of direct dependencies (i.e. directed connections) of the process model. For the sake of simplicity, whenever there is no ambiguity on the process P, we assume $\mathcal{D}$ as a synonym of $\mathcal{D}(P)$. For example, the set $\mathcal{D}$ of the process model depicted in Figure 1(a) is: $\mathcal{D} = \{A \to B, A \to C, B \to D, C \to D, D \to E\}$. We assume to have the possibility to "replay" activities of traces on the process model (e.g. [5]).

Given an event $e \in E$, such that $e = (a, u)$, let's define the projection operators $\pi_A(e) = a$ and $\pi_U(e) = u$. Moreover, let us define the operator $U_a(L)$ as:

$$U_a(L) = \{\pi_U(e) \mid \exists_{\mathbf{e} \in L}\ e \in \mathbf{e}\ \wedge\ \pi_A(e) = a\}.$$

Given a dependency $a \to b \in \mathcal{D}$, it is possible to define the set of couples of originators $U_{a \to b}(L)$:

$$U_{a \to b}(L) = \{(\pi_U(e_i), \pi_U(e_j)) \mid \text{the replay identifies a dependency of } a \to b \text{ mapped to } e_i \text{ and } e_j\}.$$

This operator returns the set of couples of originators, in the log L, that performs the dependency $a \to b$.

Similar operators are $U^a_{a \to b}(L)$ and $U^b_{a \to b}(L)$. They can be used to get originators of activity a or b, when they are involved in the dependency $a \to b$:

$$U^a_{a \to b}(L) = \{u_i \mid (u_i, u_j) \in U_{a \to b}(L)\},$$
$$U^b_{a \to b}(L) = \{u_j \mid (u_i, u_j) \in U_{a \to b}(L)\}.$$

On all these sets, it is possible to apply the classical relational algebra operators [6]. For example, using the selection operator we define $\sigma_{=}(U_{a \to b}(L)) = \{(u_i, u_j) \mid (u_i, u_j) \in U_{a \to b}(L)\ \wedge\ u_i = u_j\}$.

For simplicity, whenever there is no ambiguity on L, we assume U_a, $U_{a \to b}$ and $U^a_{a \to b}$ as a synonyms of $U_a(L)$, $U_{a \to b}(L)$ and $U^a_{a \to b}(L)$, respectively.

Given the sets $U_a(L)$, $U_{a \to b}(L)$, and $U^a_{a \to b}(L)$, we want to define the multisets [7] $\mathcal{U}_a(L)$, $\mathcal{U}_{a \to b}(L)$, and $\mathcal{U}^a_{a \to b}(L)$ which take into account the frequency of the originators in L:

$$\mathcal{U}_a(L) = \langle U_a(L), f_{U_a} \rangle \qquad \mathcal{U}_{a \to b}(L) = \langle U_{a \to b}(L), f_{U_{a \to b}} \rangle$$
$$\mathcal{U}^a_{a \to b}(L) = \langle U^a_{a \to b}(L), f_{U^a_{a \to b}} \rangle$$

where f_{U_a}, $f_{U_{a \to b}}$, and $f_{U^a_{a \to b}}$ are *multiplicity functions*, which indicate the number of times that each element of the corresponding set is observed in L. For example, given $u \in \mathcal{U}_a(L)$, $f_{U_a}(u)$ returns the number of times that the originator u performs activity a in L. In this work, the cardinality of a multiset $\mathcal{M} = \langle M, f_M \rangle$ is defined as $|\mathcal{M}| = \sum_{m \in M} f_M(m)$. The intersection of two multisets $\mathcal{M}_1 = \langle M_1, f_{M_1} \rangle$ and $\mathcal{M}_2 = \langle M_2, f_{M_2} \rangle$ is defined as the intersection of the two

sets M_1 and M_2 and the multiplicity function is defined as the minimum between the multiplicity values:

$$\mathcal{M}_1 \cap \mathcal{M}_2 = \langle M_1 \cap M_2, \min\{f_{M_1}(x), f_{M_2}(x)\}\rangle.$$

In this context, we will also consider the sum of multisets. Given $\mathcal{M}_1 = \langle M_1, f_{M_1}\rangle$ and $\mathcal{M}_2 = \langle M_2, f_{M_2}\rangle$, the sum is defined as:

$$\mathcal{M}_1 \uplus \mathcal{M}_2 = \langle M_1 \cup M_2, f_{M_1}(x) + f_{M_2}(x)\rangle.$$

For the sake of simplicity, we will omit L whenever there is no ambiguity (e.g, $\mathcal{U}_a$ instead of $\mathcal{U}_a(L)$). Moreover, the notation $\mathcal{M} = \{a^x, b^y\}$ identifies the multiset where a has multiplicity x and b has multiplicity y.

The selection operator σ_θ can be used also on multisets. For example, $\sigma_=(\mathcal{U}_{a\to b}) = \langle \sigma_=(U_{a\to b}), f_{U_{a\to b}}\rangle$ (where the multiplicity function is defined only on elements of the set $\sigma_=(U_{a\to b})$).

The problem we try to solve is to find a partition [8] $\mathbf{R} \subset \mathcal{P}(A)$[1] of the set of activities A, given a log L and the original process P. From the business point of view, we are requiring that each activity belongs to exactly one role. The partition $\mathbf{R}$ identifies the set of roles of the process. In this paper, the term "partition of activities" and "role" are used as synonyms. It is possible to see our task as a "clustering problem" [9]: we group our elements (i.e. activities) into clusters (i.e. roles), according to the features they share (i.e. originators performing activities belonging to the same role). Moreover, the rationale behind our approach is the same of clustering algorithms: we aim at achieving partitions with high intra-cluster and low inter-cluster similarities (i.e. each role is characterized by a specific set of users).

Let $|L|$ be the size of the log, i.e., the number of traces it contains. Given a log L and an originator $u \in U$, it is possible to define $|L|^u$ as:

$$|L|^u = \sum_{\mathbf{e}\in L} \sum_{i=1}^{|\mathbf{e}|} |\ \{e_i \mid \pi_U(e_i) = u\}|.$$

In other words, $|L|^u$ returns the number of times that originator u executes activities in L. A similar measure, which also takes into account the role is $|L|^u_R$, where R is a role:

$$|L|^u_R = \sum_{\mathbf{e}\in L} \sum_{i=1}^{|\mathbf{e}|} |\{e_i \mid \pi_A(e_i) \in R \ \wedge\ \pi_U(e_i) = u\}|.$$

Finally, given a log L and a partition R, it is possible to define the multiset of originators involved in the role as:

$$\mathcal{U}_R(L) = \biguplus_{a\in R} \mathcal{U}_a(L).$$

As presented in Section II, approaches for the identification of the handover of work between originators exist; however, this work proposes an approach to point out *handover of roles* and therefore the identification of roles themselves. This operation is based on activity originators. Specifically, we assume that, under ideal scenarios, there is a clear distinction of originators performing activities belonging to different roles. However, it is really difficult to observe such clear distinction in business environments (i.e., originators are involved in several roles) and thus we need to resort to a metric to measure the degree of handover between roles. This and how to define a role are the topics covered by the next section.

[1] $\mathcal{P}(A)$ identifies the powerset of A.

IV. Rules for Handover of Roles

As stated in the previous section, the identification of business roles, as presented in this work, assumes that an activity is not allowed to belong to two roles at the same time. Let us recap: given a process P and the dependency $a \to b \in \mathcal{D}(P)$, $\mathcal{U}^a_{a\to b}(L)$ is the multiset of originators (with frequencies) that perform the activity a (as part of the dependency $a \to b$) in the log L; and $\mathcal{U}_{a\to b}(L)$ identifies the set of couples of originators (with frequencies) performing a followed (possibly after some time) by b. Given a dependency between two activities we present a couple of rules which, combined, indicate if there is handover of role between the two activities. Specifically, the combination of rules indicates a measure of the expectation of handover between roles.

A. Rule for Strong No Handover

The first rule is used to identify the absence of handover of role. In this case, given the multiset $\mathcal{U}_{a\to b}$ for a dependency between two activities $a \to b$, the idea is to check if there are couples $(u, v) \in \mathcal{U}_{a\to b}$ such that $u = v$. If this is the case, it means that there is an originator performing both a and b. As stated previously, we assume that one person hardly holds more than one role; thereby there is no handover of role between subsequent activities performed by the same originator.

B. Rule for No Handover

The previous rule applies only on very specific situations. More generally, given a dependency $a \to b \in \mathcal{D}$ if the two sets of originators are equal, i.e. $U_a = U_b$, we assume there is no handover of role. This rule can be seen as a weaker version of the previous one: there are originators interchangeably performing a and b. On the contrary, if $U_a \cap\ U_b = \emptyset$ then, each activity has a disjoint set of originators and this is the basic assumption to have handover of role between a and b.

In typical business scenarios, however, it is very common to have border-line situations, and that is why a "boolean-valued" approach is not feasible. In the following, we propose a metric to capture the degree of handover of role between two activities.

C. Degree of No Handover of Roles

Given a process P a dependency $a \to b \in \mathcal{D}(P)$, and the respective multisets $\mathcal{U}^a_{a\to b}$, $\mathcal{U}^b_{a\to b}$ and $\mathcal{U}_{a\to b}$, it is possible to define the degree of no handover of role w_{ab}, that captures the rules above mentioned:

$$w_{ab}(L) = \frac{|\mathcal{U}^a_{a\to b}(L) \cap\ \mathcal{U}^b_{a\to b}(L)| + |\sigma_=(\mathcal{U}_{a\to b}(L))|}{|\mathcal{U}^a_{a\to b}(L)| + |\mathcal{U}^b_{a\to b}(L)|}, \quad (1)$$

The numerator of this equation considers the intersection of the two multisets of originators (to model no handover) plus the number of originators that perform both activities a and b (to model strong no handover). These weights are divided by the sum of the sizes of the two multisets of originators.

By definition, Equation (1) identifies the absence of handover of role. Specifically, it assumes values in the closed interval $[0, 1]$, where 1 indicates there is no handover of roles and 0 indicates handover. Since the ideal case (i.e., completely disjoint sets of originators for each role) is very unlikely, we propose to use a threshold τ^w on the value w_{ab}. If $w_{ab} > \tau^w$, then there is no handover of roles; otherwise the handover occurs. A partition of the activities can then be obtained by removing from the process model all the dependencies which corresponds to handovers: connected activities are in the same element of the partition (see Fig. 2).

Example 1: Given a process P, a log L, and the dependency $a \to b \in \mathcal{D}(P)$, assume that:

- $\mathcal{U}^a_{a\to b}(L) = \{u^1_1, u^1_2, u^1_3\}$,
- $\mathcal{U}^b_{a\to b}(L) = \{u^1_1, u^1_2, u^1_3\}$, and
- $\mathcal{U}_{a\to b}(L) = \{(u_1, u_1)^1, (u_2, u_2)^1, (u_3, u_3)^1\}$.

The value $w_{ab}(L) = 1$ strongly indicates there is no handover of role in this case. In fact, as the set $\mathcal{U}_{a\to b}(L)$ suggests, the same originator is observed performing both a and b several times.

Example 2: Let's now consider a scenario completely different from Example 1. Given a process P, a log L, and the dependency $a \to b \in \mathcal{D}(P)$, assume that:

- $\mathcal{U}^a_{a\to b}(L) = \{u^1_1, u^1_2, u^1_3\}$,
- $\mathcal{U}^b_{a\to b}(L) = \{u^1_4, u^1_5, u^1_6\}$, and
- $\mathcal{U}_{a\to b}(L) = \{(u_1, u_4)^1, (u_2, u_5)^1, (u_3, u_6)^1\}$.

The value $w_{ab}(L) = 0$ strongly indicates the presence of handover of role. It can be seen that the two sets of originators do not share any person and, based on our assumptions, this is a symptom of handover.

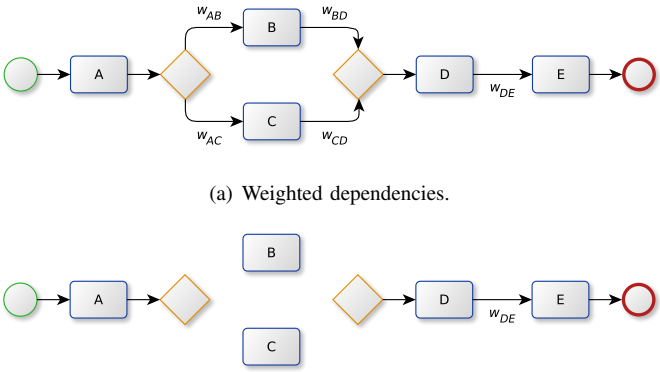
Example 3: Consider now a third example, in the middle between Ex. 1 and 2. Given a process P, a log L, and the dependency $a \to b \in \mathcal{D}(P)$, assume that:

- $\mathcal{U}^a_{a\to b}(L) = \{u^1_1, u^1_2, u^1_3\}$,
- $\mathcal{U}^b_{a\to b}(L) = \{u^1_1, u^1_2, u^1_4\}$, and
- $\mathcal{U}_{a\to b}(L) = \{(u_1, u_1)^1, (u_2, u_4)^1, (u_3, u_2)^1\}$.

In this case, $w_{ab}(L) = 0.5$ so there is no clear handover. Looking at the originator sets, u_1 performs subsequently a and then b, in one case. Moreover, u_2 is observed performing both a and b but not on the same process instance. In this example, it turns out to be fundamental the value of the threshold τ^w, in order to decide if handover of role occurs.

D. Merging Roles

As mentioned in the introductory part, the approach presented in this paper is based on two steps: the first step identifies handover of roles (through the metric w_{ab}) which induces a partition of activities, i.e. roles. Clearly, this way of performing the partitioning is too aggressive: if the control-flow "comes back" to roles already discovered, the handover does not entail the creation of a new role. The aim of the second step is to merge partitions that are supposed to represent the same role.



(a) Weighted dependencies.

(b) Removed dependencies associated to handover of roles.

Fig. 2. Process model of Fig. 1(a) with weights associated to every dependency (*top*), and after the dependencies associated to handover of roles are removed (*bottom*). Activities are thus partitioned into the subsets $\{A\}$, $\{B\}$, $\{C\}$, $\{D, E\}$.

Given a process P and a log L, the first step generates a partitioning $\mathbf{R}$ of the activities. In order to merge some roles, we propose a metric which returns the merging degree of two partitions. Given two roles $R_i, R_j \in \mathbf{R}$:

$$\rho_{R_i R_j}(L) = \frac{2|\mathcal{U}_{R_i}(L) \cap \mathcal{U}_{R_j}(L)|}{|\mathcal{U}_{R_i}(L)| + |\mathcal{U}_{R_j}(L)|}. \quad (2)$$

The basic idea of this metric is the same as presented in Equation (1), i.e., to measure the amount of shared originators between the two roles. This metric produces values on the closed interval $[0, 1]$ and, if activities of the two partitions are performed by the same originators, the values of the metric is 1 (and therefore the two roles are supposed to be the same and merged). If the roles have no common originators, then the value of ρ is 0 and the roles are kept separated.

Due to the blurry situations that are likely in reality, a threshold τ^ρ is employed: if $\rho_{R_i R_j}(L) > \tau^\rho$ then R_i should be merged with R_j; otherwise they are considered distinct roles.

V. Algorithm Description

In this section, we give some algorithmic details concerning the two previously described steps. We do this with the help of the process described in Figure 1(a). Moreover, we give an algorithm to generate all "plausible" partitions of activities (sets of candidate roles).

A. Step 1: Handover of Roles Identification

The first step of our approach consists in the identification of the partitions induced by every handover of role. Please note that, in our context, an handover of role may occur only when the work passes from one activity to another (i.e., dependencies between activities of the process).

To achieve our goal, given a process P, the algorithm starts by extracting all the dependencies $\mathcal{D}(P)$. After that,

Algorithm "Roles Aggregation"

Require: Log L; a set of roles $\mathbf{R}$; and threshold $\tau^{\rho} \in [0, 1]$

1: **repeat**
2: $\rho_{max} \leftarrow \max_{(R_i,R_j)\in\mathbf{R}\times\mathbf{R}} \rho_{R_iR_j}(L)$
3: $R_{\rho_{max}} \leftarrow \arg\max_{(R_i,R_j)\in\mathbf{R}\times\mathbf{R}} \rho_{R_iR_j}(L)$ ▷ Maximals
4: **if** $\rho_{max} \geq \tau^{\rho}$ **then**
5: Choose $(R_i, R_j) \in R_{\rho_{max}}$ ▷ Selection is performed considering the couple that maximizes the number of merged originators, if necessary the number of merged activities and, finally, the lexicographical order of role activities.
6: $\mathbf{R} \leftarrow (\mathbf{R} \setminus \{R_i, R_j\}) \cup \{R_i \cup R_j\}$ ▷ Merge R_i and R_j
7: **end if**
8: **until** no merge is performed
9: **return** $\mathbf{R}$

Fig. 3. Algorithm to perform the aggregation of roles (i.e. "Step 2").

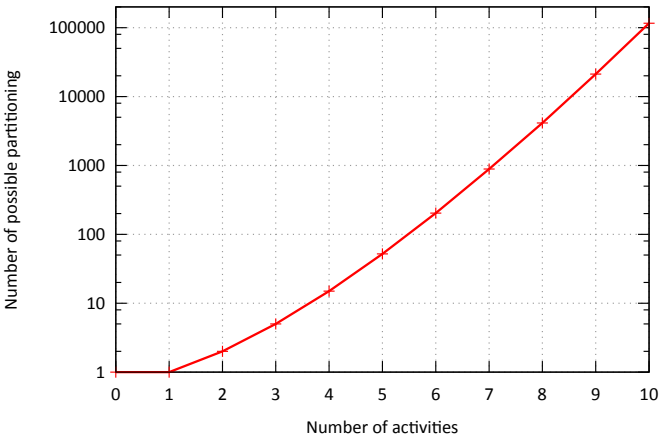


Fig. 4. Representation of the growth of the number of possible partitioning, given the number of elements of a set.

every dependency is weighted using Equation (1) (the result is reported in Figure 2(a)). At this point, we apply a threshold τ^{w}. Specifically, we consider a particular dependency as handover of role only if its weight is less or equal to τ^{w}. Every time an handover is observed, the corresponding dependency is removed from the process.

Let's consider again the example process of Fig. 1(a) and the weights of Fig. 2(a). Let's assume $w_{ab} \leq \tau^{w}$, $w_{ac} \leq \tau^{w}$, $w_{bd} \leq \tau^{w}$, $w_{cd} \leq \tau^{w}$ and $w_{de} > \tau^{w}$. Fig. 2(b) reports the process obtained after handover of roles have been removed.

At the end of the first step, four roles have been identified: $\{A\}$, $\{B\}$, $\{C\}$, and $\{D, E\}$. These roles correspond to the activities of the connected components [10] of Fig. 2(b).

B. Step 2: Roles Aggregation

As stated previously, the first step of the approach identifies roles which may be too fine grained. For example, in Fig. 2(b) each connected component represents a role, however, as Fig. 1(b) shows, we actually want A in the same role of D and E, and we want B together with C. In this step, we use Equation (2) to evaluate if any couple of roles may be merged.

Figure 3 proposes the pseudocode of the procedure used in this phase. It requires, as input, a log L, a set of roles (i.e., a partitioning of activities) $\mathbf{R}$ and a value for the threshold τ^{ρ}. First of all, the algorithm finds the best pairs of roles that can be merged (line 3), i.e., pairs with maximal ρ. If the best value of ρ is above the threshold τ^{ρ}, it means that it is possible to merge two roles. However, several pairs may have the same maximal ρ. The criterion to select just one pair is to consider the roles that maximize the number of affected originators. If there are several pairs with identical ρ values and number of affected originators, we choose the pair that maximizes the number of merged activities. If we still have more than one pair, we just pick the first pair according to lexicographical order of contained activities (line 5). The two selected candidate roles are then merged. The same procedure is repeated until no more roles are merged (line 8), i.e., there is no pair with value of ρ above the threshold τ^{ρ}. Finally, the modified set of roles is returned (line 9).

C. Generation of Candidate Solutions

The approach, as presented so far, requires the configuration of two thresholds, i.e. τ^{w} and τ^{ρ}. Little variations in configuration of these parameters may lead to very different roles. To tackle this problem, we think it might be interesting to extract all the significant partitioning and propose them to the user. Given the set A of tasks, the number of possible partitions is identified by the Bell number [8]. This quantity, given n as the size of the set is recursively defined as:

$$B(n) = \sum_{t=0}^{n-1} \binom{n-1}{t} B(t)$$

Figure 4 presents the explosion of the number of possible partitioning, given the number of elements of a set.

By construction, the proposed approach requires two parameters: τ^{w} and τ^{ρ}. The values of these two threshold is required to be in the interval $[0, 1]$; however, it can be seen that only a finite number of values produces different results.

As example, if we consider τ^{w}, it is used to remove edges from the original process. Since the number of edges of a process is finite, there is a finite number of values of τ^{w} that splits activities of the process. The same observation can be used to enumerate the possible values of τ^{ρ}.

The algorithm described in Figure 5 proposes an approach which automatically extracts all the significant configurations of τ^{w} and τ^{ρ} and returns such set of solutions. Specifically, line 2 collects all the significant values of τ^{w}. All these values are used to remove the handover of roles (line 6-10). Line 12, given the partitioning just obtained, it generates the set of all significant values for τ^{ρ}, which are considered for the computation of step 2 (line 15). The returned result consists of a set with all the significant partitions (with respect to the log L) that can be extracted.

The algorithm proposed in Fig. 5 has a worst-case complexity which is $O(n^3)$, where n is number of edges of the given process. In fact, it is possible that each dependency of the process has a different weight w_{ab}. The same situation

Algorithm "Roles Selection"

Require: Process P; and a log L

$\mathbf{S} \leftarrow \emptyset$ ▷ Set of final solutions
$T^w \leftarrow \{w_{ab}(L) \mid a \rightarrow b \in \mathcal{D}(P)\}$
for all $\tau^w \in T^w$ **do** ▷ Tries all significant values for τ^w
  Copy the process P in P'
  ▷ Step 1
  for all $a \rightarrow b \in \mathcal{D}(P)$ **do**
    if $w_{ab}(L) \leq \tau^w$ **then**
      Remove dependency $a \rightarrow b$ from P'
    end if
  end for
  $\mathbf{R} \leftarrow$ set of activities in connected components of P'
  $T^\rho \leftarrow \{\rho_{R_i R_j}(L) \mid R_i, R_j \in \mathbf{R}\}$
  for all $\tau^\rho \in T^\rho$ **do** ▷ Tries all significant values for τ^ρ
    ▷ Step 2
    $\mathbf{R}_{final} \leftarrow$ **Roles Merger** (L, $\mathbf{R}$, τ^ρ) ▷ See Fig. 3
    $\mathbf{S} \leftarrow \mathbf{S} \cup \{\mathbf{R}_{final}\}$ ▷ Consider the new solution
  end for
end for
return $\mathbf{S}$

Fig. 5. Complete algorithm to automatically find all different partitioning of activities, given a log, and a process model.

may happen when considering ρ_{AB}: it is possible to have n clusters from step 1, and each pair of them can have a different value of ρ_{AB}. However, it is important to note that, typically, n is relatively small and, more importantly, is independent from the given log. In particular, it is necessary to analyze the log (linear complexity, with respect to the number of events it contains), but this operation is performed only once: all the other activities (reported in Fig. 5) can use the already collected statistics.

It is possible to sort the set of partitions according to the number of roles. This ordered set is then proposed to the final user. In this way, the user will be able to explore all the significant alternate definitions of roles.

D. Partition Evaluation

A possible way to evaluate the discovered partitions, is to use the concept of entropy [11]. In this context, we propose our measure. Specifically, given $\mathbf{R}$ as the current partition, i.e., set of roles (each role is a set of activities), U as the set of originators, and L as a log, we define an entropy measure of the partition as:

$$H(\mathbf{R}, L) = \sum_{u \in U} \sum_{R \in \mathbf{R}} -\frac{|L|_R^u}{|L|^u} \log_2 \left(\frac{|L|_R^u}{|L|^u} \right). \quad (3)$$

Let us recall that $|L|_R^u$ is defined as the number of times that activities belonging to the role R, and performed by user u, are observed in L; and that $|L|^u$ is defined as the number of activities executed by originator u in the log L.

This measure is zero if each originator is involved in one and only one role. Otherwise, the measure increases with the degree of mixture of contribution of originators to multiple roles.

VI. Experiments

The approach presented in this paper has been implemented in the ProM 6.2 [12] toolkit and will freely be available soon. We performed several tests against an artificial dataset.

In our datasets, we have the target partitioning (i.e. the expected roles) and, given a log, our goal is to discover those roles. To evaluate our results we compare the target roles with the extracted ones and we use a measure inspired by purity [13]. Let us recall that A represents the set of activities (or tasks) of the process and that a role is a set of activities. $|R|$ is the number of activities contained in the role R. Given the target set of roles $\mathbf{R}_t$ and the discovered one $\mathbf{R}_d$, our degree of similarity is defined as:

$$\textit{similarity} = \frac{1}{|\mathbf{R}_d|} \sum_{R_d \in \mathbf{R}_d} \max_{R_t \in \mathbf{R}_t} \frac{2|R_d \cap R_t|}{|R_d| + |R_t|}.$$

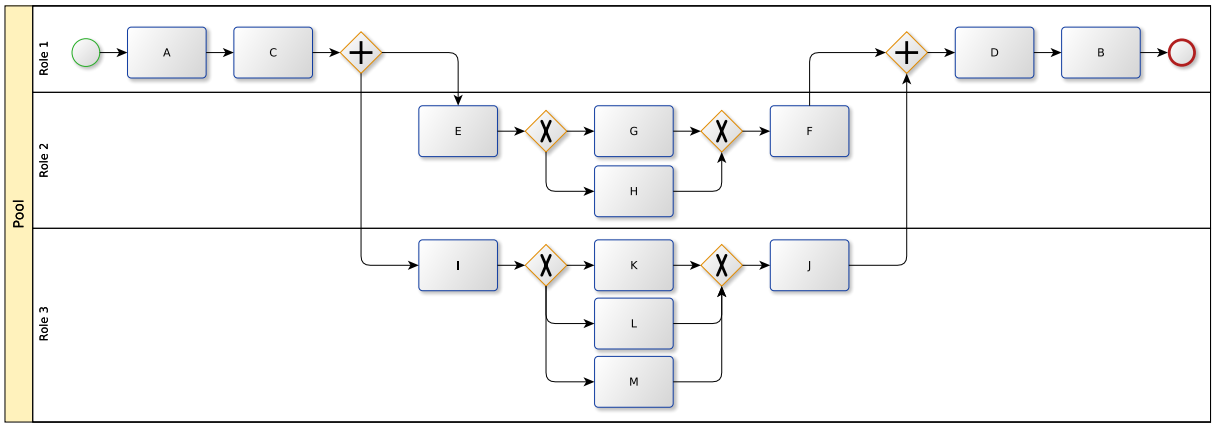
The idea behind this formulation is that if the partitioning discovered is equal to the target, the similarity value is 1, otherwise it decreases.

Two artificial processes have been created with PLG [14]. These processes, shown in Figure 6, have been simulated 1000 times. However, due to the lack of PLG in simulating the organizational perspective, we prepared a second ProM package, which is able to decorate a given log with information on the originators. This package requires some configuration: first of all, it is necessary to set the number of roles and the number of originators. After that, a distribution of activities among roles is required, and it is necessary to assign each originator to one or more roles. The package will decorate the log by randomly choosing, for each activity, an originator belonging to the same role of the activity itself. By default, each originator of a role is equally likely to be selected, however, a simple procedure to specify *ad-hoc* probabilities is available as well.

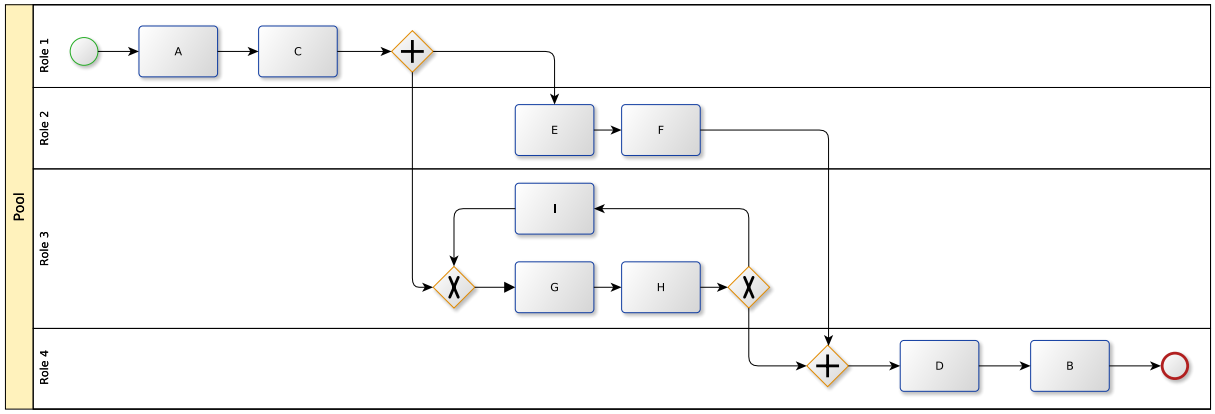
A. Model 1

Model 1 (Figure 6 (a)) contains 13 activities divided over 3 roles. A peculiarity of this process is that the workflow starts with activities belonging to "Role 1" and finishes with other activities belonging to the same "Role 1". This process have been simulated to generate five different logs:

- one with exactly one originator per role;
- another with exactly two originators per role;
- the third log is similar to the second but is also includes a "jolly": an originator performing all activities;
- the fourth log contains three originators; all of them are involved on all activities, however, each role has a "leader". Given a role, an activity is executed by its leader with probability 0.5, otherwise all other originators are equally likely;
- the last log has 6 originators performing all the activities with a leader for each role (with the same probabilities of the previous case).



(a) Model 1.



(b) Model 2.

Fig. 6. Process models generated for the creation of the artificial dataset.

B. Model 2

Model 2 (Figure 6 (b)) is composed by 9 activities and 4 roles. In this case, the process also has a loop of activities within "Role 3". This process has been simulated to generate 3 logs:

- one with exactly one originator per role;
- another with exactly two originators per role;
- the last one with 8 originators, all of them involved in all the activities, with one "leader" per role (with same probabilities of last logs of Model 1).

C. Results

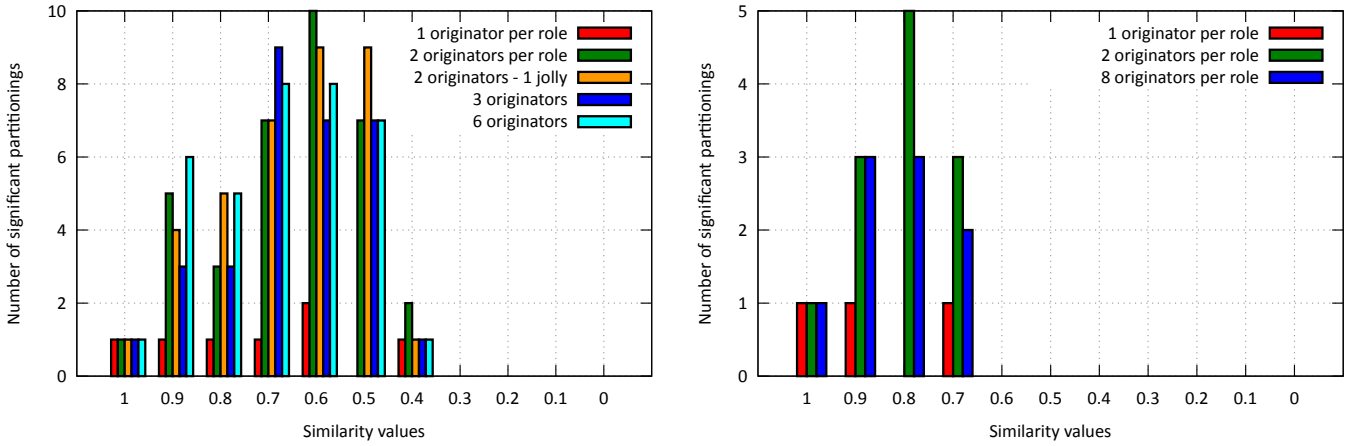
The first results are presented in Table I. Specifically, for each log, the number of different partitions is reported. Please note that this number is always relatively small and, in the worst cases (log with six users and log with jolly of the first model), we have 36 different partitionings.

Figure 7 proposes, for the two models, the distribution of the partitions according to the corresponding similarity measure (with respect to target roles). Concerning the logs of Model 1, all the partitions have similarity values very high, most of them are concentrated on the interval $[1, 0.5]$. In the case of Model 2, most of partitions lay on the interval $[1, 0.7]$. It is very important to note that in all cases the system extracts the target roles (i.e. partition with similarity 1).

TABLE I
THIS TABLE REPORTS THE RESULTS, FOR THE TWO MODELS, IN TERMS OF NUMBER OF SIGNIFICANT DIFFERENT PARTITIONS DISCOVERED.

Logs	**No. of partitions**
(a) **Model 1**	
1 user per role	6
2 users per role	34
2 users per role – 1 jolly	36
3 users	31
6 users	36
(b) **Model 2**	
1 user per role	3
2 users per role	12
8 users	9

The purpose of the last experiment is to evaluate the entropy measure. Specifically, for each log, we ranked all partitions according to the corresponding entropy measure. After that, we verified the position of the target partition. Results are reported



(a) Results for Model 1.

(b) Results for Model 2.

Fig. 7. Results, for the two models, in terms of number of significant partitioning with respect to the purity value, reported in bin of width 0.1.

TABLE II
THIS TABLE REPORTS, FOR EACH LOG, THE RANK OF THE TARGET PARTITION. RANKING IS BASED ON THE ENTROPY VALUE.

Logs	**Rank of target partition**
(a) **Model 1**	
1 user per role	1
2 users per role	1
2 users per role – 1 jolly	4
3 users with leader	4
6 users with leader	12
(b) **Model 2**	
1 user per role	1
2 users per role	1
8 users with leader	5

in Table II. As you can see, whenever there is no "confusion" (i.e. one originator is involved in exactly one role), the entropy measure suggests the desired partition (i.e. it is in first place). Instead, when the same originator performs several roles, the confusion increases and it is harder, for our entropy measure, to correctly identify the target partition (i.e. the target partition is not in first place).

VII. Conclusions and Future Work

In this paper we considered the problem of enhancing a process model with information about roles. Specifically, we aimed at discovery a partitioning of activities. This activity is performed by taking into account originators and which activities they perform. Measures of handover of roles are defined and employed; and an approach to automatically extract only the significant partitionings is shown too.

As future work, we would like to improve the entropy measure so to have a reliable approach to the automatic selection of the solution. Moreover, we think it will be very useful to conduct a deeper analysis on real datasets, in order to point out the actual capacity of our metrics (i.e., w_{ab} and $\rho_{r_i r_j}$) to discover roles.

Acknowledgement

Authors would like to thank the supporters of this work: Siav S.p.A. and the Eurostars-Eureka project PROMPT (E!6696).

References

[1] IEEE Task Force on Process Mining, "Process Mining Manifesto," in *Business Process Management Workshops*, F. Daniel, K. Barkaoui, and S. Dustdar, Eds. Springer-Verlag, 2011, pp. 169–194.

[2] W. M. P. van der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[3] M. Song and W. M. P. van der Aalst, "Towards comprehensive support for organizational mining," *Decision Support Systems*, vol. 46, no. 1, pp. 300–317, Dec. 2008.

[4] W. M. P. van der Aalst, H. A. Reijers, and M. Song, "Discovering Social Networks from Event Logs," *Computer Supported Cooperative Work (CSCW)*, vol. 14, no. 6, pp. 549–593, Oct. 2005.

[5] W. M. P. van der Aalst, A. Adriansyah, and B. van Dongen, "Replaying History on Process Models for Conformance Checking and Performance Analysis," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 182–192, Mar. 2012. [Online]. Available: http://doi.wiley.com/10.1002/widm.1045

[6] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 6th ed. Addison-Wesley, 2010.

[7] A. Syropoulos, "Mathematics of Multisets," in *Multiset Processing*, C. Calude, G. Paun, G. Rozenberg, and A. Salomaa, Eds. Springer Berlin / Heidelberg, 2001, pp. 347–358.

[8] R. A. Brualdi, *Introductory Combinatorics*, 5th ed. Pearson Prentice Hall, 2009.

[9] I. H. Witten, F. Eibe, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed. Morgan Kaufman, 2011.

[10] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*, 2nd ed. The MIT Press, Sep. 2001.

[11] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423 & 623–656, 1948.

[12] E. H. M. W. Verbeek, J. Buijs, B. van Dongen, and W. M. P. van der Aalst, "ProM 6: The Process Mining Toolkit," in *BPM 2010 Demo*, 2010, pp. 34–39.

[13] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, 1st ed. Cambridge University Press, Jun. 2008, vol. 35, no. 2.

[14] A. Burattin and A. Sperduti, "PLG: a Framework for the Generation of Business Process Models and their Execution Logs," in *Business Process Management Workshops (BPI)*. Hoboken, New Jersey, USA: Springer Berlin Heidelberg, 2010, pp. 214–219.