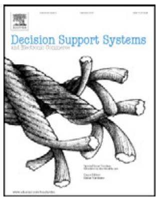
Decision Support Systems 50 (2011) 715–731

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss



A new role mining framework to elicit business roles and to mitigate enterprise risk

Alessandro Colantonio [a,b,*], Roberto Di Pietro [b], Alberto Ocello [a], Nino Vincenzo Verde [b]

[a] *Engiweb Security, Roma, Italy*
[b] *Università di Roma Tre, Dipartimento di Matematica, Roma, Italy*

ARTICLE INFO

Available online 19 August 2010

Keywords:
RBAC
Role engineering
Role mining
Risk management
Clustering coefficient

ABSTRACT

Role-based access control (RBAC) allows to effectively manage the risk derived from granting access to resources, provided that designed roles are business-driven. *Role mining* represents an essential tool for role engineers, but existing techniques are not able to elicit roles with an associated clear business meaning. Hence, it is difficult to mitigate risk, to simplify business governance, and to ensure compliance throughout the enterprise. To elicit meaningful roles, we propose a methodology where data to analyze are decomposed into smaller subsets according to the provided business information. We introduce two indices, *minability* and *similarity*, that drive the decomposition process by providing the expected complexity to find roles with business meaning. The proposed methodology is rooted on a sound theoretical framework. Moreover, experiments on real enterprise data support its effectiveness.

1. Introduction

Access control is a cornerstone of enterprise risk and security management. It represents the process of mediating requests to data and services maintained by a system, and determining whether the requests should be granted or denied [11]. It is the responsibility of an access control system to ensure that only users with legitimate credentials are granted permissions to access requested resources. Hence, in an access control model the risk factor of illegitimate credentials is eliminated by construction [2]. Significant research has focused on providing formal representations of access control models. Among all models proposed in the literature, *Role-Based Access Control* (RBAC) [1] is certainly the most adopted by medium- to large-size organizations, greatly due to its simplicity: a role can be seen as a set of permissions; users, in turn, are assigned to appropriate roles based on their responsibilities and qualifications. As a result, RBAC offers great benefits to business users. A role represents a job function or a title established for a set of users within an organization. Thus, the adoption of RBAC makes it easier to define security policies by business users [16]. RBAC also implements the appropriate security engineering principles to enforce risk reduction, such as separation of duties (SoD) and least privilege [2]. Further, the use of roles minimizes system administration effort due to the reduced number of relationships required to relate users to permissions [5].

Despite the benefits related to deploying role-based access control systems, many organizations are reluctant to adopt them, since there are still some important issues that need to be addressed. In particular, the model must be customized to capture the needs and functions of the organization. In an ideal RBAC environment, we expect roles to be well defined so that role definitions are formed with strict role boundary rules in order to enforce all the required enterprise security policies. Unfortunately, where RBAC is deployed, this rarely happens, thus leading to role misuse [2]. For this purpose, the *role engineering* discipline [9] has been introduced. However, choosing the best way to design a proper set of roles is still an open problem. Various approaches to role engineering have been proposed, which are usually classified as: *top-down* and *bottom-up*. The former requires a deep analysis of business processes to identify which access permissions are necessary to carry out specific tasks. The latter seeks to identify de facto roles embedded in existing access control information. Since bottom-up approaches usually resort to data mining techniques, the term *role mining* is often used as a synonym for bottom-up. To maximize benefits, bottom-up should be used in conjunction with top-down, leading to an *hybrid* approach. As a matter of fact, top-down may ignore existing permissions and exceptions, whereas bottom-up may not consider the business functions of an organization [19].

The bottom-up approach has attracted researchers, since it can be easily automated [22]. Indeed, companies which plan to go for RBAC usually find themselves in the situation of having a collection of several legacy and standard security systems on different platforms that provide "conventional" access control [20]. Thus, role mining is the application of data mining techniques to generate roles from the

* Corresponding author.
E-mail addresses: alessandro.colantonio@eng.it, colanton@mat.uniroma3.it (A. Colantonio), dipietro@mat.uniroma3.it (R. Di Pietro), alberto.ocello@eng.it (A. Ocello), nverde@mat.uniroma3.it (N.V. Verde).
URLs: http://ricerca.mat.uniroma3.it/users/colanton/ (A. Colantonio), http://ricerca.mat.uniroma3.it/users/dipietro/ (R. Di Pietro), http://ricerca.mat.uniroma3.it/users/nverde/ (N.V. Verde).

0167-9236/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2010.08.022

access control information of this collection of systems. Several works prove that the role mining problem is reducible to many other well-known NP-hard problems, such as clique partition, binary matrix factorization, bi-clustering, graph vertex coloring [6,8,32] to cite a few. However, on one hand the slavish application of standard data mining approaches to role engineering might yield roles that are merely a set of permissions, namely with no connection to the business practices. On the other hand organizations are unwilling to deploy roles they cannot bind to a business meaning [5]. Indeed, such roles could have some difficulties in being inserted within the risk management framework in use within the organization. In such a case, risks are incurred to the system by users that are authorized to use their access right in an incorrect manner [2]. Moreover, when hundreds of thousands of existing user–permission assignments need to be analyzed, the number of candidate roles might be so high that trying to assign a business meaning to each of them is often impracticable. The number of candidate roles may also grow because of the "noise" within the data—namely, permissions exceptionally or accidentally granted or denied. In such a case, classical role mining algorithms discover multiple small fragments of the true role, but missing the role itself [7]. This increases the risk of designing roles that do not capture the actual business needs of the organization.

Only a few recent works value business requirements in role mining [3,5] by proposing a measure for the business meaning of roles. However, it is difficult to introduce this metric in existing role mining approaches currently found in the literature. An alternative way of leveraging business-related information to offer meaningful candidate role-sets may be by restricting the analysis to sets of data that are homogeneous from an enterprise perspective. For instance, let us suppose that a partial or coarse-grained top-down analysis identifies a certain set of users that perform the same tasks, but the analysis lacks the knowledge of which permissions are required for the execution of these tasks. In this scenario, by restricting role mining techniques to these users only—instead of analyzing the organization as a whole—, the elicited roles will be related to such tasks. Thus, it will likely be easier to assign a business meaning to the results obtained from the bottom-up approach. Moreover, by grouping users that perform similar tasks together first, and then analyzing them separately, eliciting roles with no business meaning can be avoided. Indeed, investigating analogies among groups of users that perform completely different tasks is far from being a good role mining strategy [5]. Further, it will be easier to manage resulting candidates roles, achieving two results: a simplification of the security policy enforcement process; and, a reduction of the risk related to unintentional/incorrect use of granted permissions through roles. Partitioning data also introduces benefits in terms of execution time of role mining algorithms. Indeed, most role mining algorithms have a complexity that is not linear with respect to the number of users or permissions to analyze [3,13,21,30]. Based on previous observations, several enterprise information may be used to decompose the role mining problem. Business processes, workflow tasks, and organization unit trees are just a few examples of business elements that can be leveraged. Usually, such information is already available in most companies before starting the role engineering task. However, when dealing with information from several sources, a few decisions have to be made about: what information can actually improve the role mining process; what level of detail is required; and, lastly, how to verify that each sub-problem is easily solvable using a data mining algorithm.

To address all the abovementioned issues, this paper proposes a methodology that helps role engineers leverage business information during the role mining process. In particular, we propose to divide the access data to analyze into smaller subsets that are homogeneous according to some business data, instead of performing a single bottom-up analysis on the entire organization. This eases the attribution of business meaning to automatically elicited roles and reduces the problem complexity, thus allowing for better enforcement of security policies and reducing the risk related to illegal accesses. In order to select the best business information that improves the subsequent role mining process, as well as to establish how deeply the data must be partitioned, two indices, referred to as *minability* and *similarity*, are identified. Minability and similarity are both rooted on sound mathematical theory. These indices are used to measure the expected complexity of analyzing the outcome of the bottom-up approaches. Leveraging these indices allows for the identification of business information that best fits with the access control data, namely the information that induces a decomposition which increases the business meaning of the roles elicited in the role mining phase and, at the same time, simplifies the analysis. This leads to a decrease in the likelihood of making errors in role management, and consequently reduces the risk of role misuse. The paper also introduces two fast probabilistic algorithms to efficiently compute such indices, making them suitable also for big organization with hundreds of thousands of users and permissions. The quality of the indices is also formally assured. Several examples illustrate the practical implications of the proposed methodology and related tools, which have also been applied on real enterprise data. Results support the quality and viability of the proposal.

The paper is organized as follows: Section 2 introduces the background required to formally describe the proposed tools and reports on related works. The adopted risk model is then described in Section 3, that also maps typical risk-related concepts to RBAC entities. Minability and similarity indices are introduced and further discussed with simple examples in Section 4. The proposed methodology, which is mainly based on these two indices, is proposed in Section 5. In Section 6 two efficient probabilistic algorithms are proposed: one to calculate similarity, the other one to calculate the minability index. Then, the viability of the proposed applications is demonstrated in Section 7, showing the results of a test on real data. Finally, Section 8 provides concluding remarks.

2. Background and related work

2.1. Role engineering

Before introducing the required formalism used to describe role engineering, we first review some concepts of the ANSI/INCITS RBAC standard [1] needed for the present analysis. For the sake of simplicity, we do not consider sessions, role hierarchies or separation of duties constraints in this paper. In particular, we are only interested in the following entities:

- *PERMS*, *USERS*, and *ROLES* are the sets of all access permissions, users, and roles, respectively;
- $UA \subseteq USERS \times ROLES$, is the set of all role–user relationships;
- $PA \subseteq PERMS \times ROLES$, is the set of all role–permission relationships.

The following functions are also provided:

- $ass\_users : ROLES \rightarrow 2^{USERS}$ to identify users assigned to a role. We consider it as derived from *UA*, that is $ass\_users(r) = \{u \in USERS | \langle u, r \rangle \in UA\}$;
- $ass\_perms : ROLES \rightarrow 2^{PERMS}$ to identify permissions assigned to a role. We consider it as derived from *PA*, that is $ass\_perms(r) = \{p \in PERMS | \langle p, r \rangle \in PA\}$.

In addition to RBAC concepts, this paper introduces other entities required to formally describe the proposed approach. In particular, we define:

- $UP \subseteq USERS \times PERMS$, the set of the existing user–permission assignments to be analyzed;
- $perms : USERS \rightarrow 2^{PERMS}$, the function that identifies permissions assigned to a user. Given $u \in USERS$, it is defined as $perms(u) = \{p \in PERMS | \langle u, p \rangle \in UP\}$;

- $users: PERMS \rightarrow 2^{USERS}$, the function that identifies users that have been granted a given permission. Given $p \in PERMS$, it is defined as $users(p) = \{u \in USERS | \langle u,p \rangle \in UP\}$.

Having introduced these entities, it is now possible to formally define the main objective of role engineering: given *UP*, *PERMS*, and *USERS*, we are interested in determining the best setting for *ROLES*, *PA*, and *UA* that covers all possible combinations of permissions possessed by users. In this context "best" means that the proposed roles should maximize the advantages offered by adopting RBAC, that is, to simplify business governance, to mitigate risk, and to ensure compliance throughout the enterprise. This can be seen as a multi-objective optimization problem [5]. As for the coverage, there is a need that for each $\langle u,p \rangle \in UP$ at least one role $r \in ROLES$ should exist such that $u \in ass\_users(r)$ and $p \in ass\_perms(r)$.

Role engineering was first illustrated by Coyne [9] through a top-down perspective. Many other authors sought to leverage business information to design roles by adopting a top-down approach such as [19,23,26]. These works represent pure top-down approaches—they do not consider existing access permissions—; hence, they do not take into account how the organization actually works. As for the bottom-up approach, Kuhlmann et al. [20] first introduced the term "role mining", trying to apply existing data mining techniques to elicit roles from existing access data. After that, several algorithms explicitly designed for role engineering were proposed [13,14,21,28,30,33]. The main limitation of these works is that they do not always lead to the optimal set of roles from a business perspective. To the best of our knowledge, the work from Colantonio et al. [3] represents the first approach that allows for the discovery of roles with business meanings through a role mining algorithm. A cost function is introduced as a metric for evaluating a "good" collections of roles. By minimizing the cost function it is possible to elicit those roles that contextually minimize the overall administration effort and fit the needs of an organization from a business perspective. Further improvements of this approach are represented by [4,5,6,8].

2.2. Role mining and graphs

The approach proposed in this paper is based on a formal correspondence between the role mining problem and selected problems from graph theory. Thus, we first need to review some graph-related concepts. A *graph* G is an ordered pair $G = \langle V_G, E_G \rangle$, where V_G is the set of vertices, and E_G is a set of unordered pairs of vertices [12]. We say that $v,w \in V_G$ are *endpoints* of the edge $\langle v,w \rangle \in E_G$. Given the subset of vertices $S\ V_G$, then the *subgraph induced* by S is the graph that has S as vertex set, and members of E_G are such that their endpoints are both in S as edge set. A *bipartite graph* $B = \langle V_B, E_B \rangle$ is a graph where the set of vertex V_B can be partitioned into two subsets V_1 and V_2 such that $\forall \langle v_1, v_2 \rangle \in E_B : v_1 \in V_1, v_2 \in V_2$.

A *clique* (of an unipartite graph G) is a subset of vertices $S\ V_G$, such that the subgraph induced by S is a *complete* graph, namely for every two vertices in S an edge connecting the two exists. A *biclique* in a bipartite graph B, also called *bipartite clique*, is a set of vertices $W_1\ V_1$ and $W_2\ V_2$ such that $\forall w_1 \in W_1, \forall w_2 \in W_2 : \langle w_1, w_2 \rangle \in E_B$. In both the unipartite and bipartite cases, we will say that a set of edges *induces a (bi)clique* if the subgraph induced by the endpoints of the edges is a (bi)clique. A *clique cover* of a unipartite graph G is a collection of cliques $S_1, \ldots, S_k$, such that for each edge $\langle u,v \rangle \in E_G$ there is some S_i that contains both u and v. A *clique partition* of a graph is a collection of cliques such that each vertex is a member of exactly one of the cliques: it is a partition of the vertices into cliques. Similarly, a *biclique cover* of a bipartite graph B is a collection of biclique $W_1, \ldots, W_k$ such that for each edge $\langle v_1, v_2 \rangle \in E_B$ there is some W_i that contains both v_1 and v_2. Thus, each edge of B is covered by at least one biclique.

An access control system configuration can be represented by a bipartite graph $B = \langle USERS \cup PERMS, UP \rangle$, where two vertices $u \in USERS$ and $p \in PERMS$ are connected by an edge if the user u is granted permission p, namely $\langle u,p \rangle \in UP$. A biclique cover of this graph B univocally identifies a *candidate role-set* [5], namely a set of roles, and for each role a set of users and permissions, such that all the user–permission assignments belonging to *UP* can be covered by at least one role. Indeed, every biclique identifies a role, and the vertices of the biclique identify the users and the permissions assigned to this role [6,13]. Starting from the bipartite graph B set up via the user–permission relations in *UP*, it is possible to construct an undirected unipartite graph G in the following way: each edge in B (i.e., a user–permission relationship of *UP*) becomes a vertex in G, and two vertices in G are connected by an edge if and only if the endpoints of the corresponding edges of B induce a biclique. To ease exposition, we define the function $biclique: UP \rightarrow 2^{UP}$ that indicates all edges in *UP* which induces a biclique together with the given edge, namely:

$$biclique(\langle u,p \rangle) = \{\langle u',p' \rangle \in UP \,|\, \langle u,p' \rangle, \langle u',p \rangle \in UP \wedge \langle u,p \rangle \neq \langle u',p' \rangle\}. \quad (1)$$

Note that a pair of edges $\omega_1 = \langle u_1, p_1 \rangle$ and $\omega_2 = \langle u_2, p_2 \rangle$ of *UP* that share the same user (that is, $u_1 = u_2$) or the same permission (that is, $p_1 = p_2$) induce a biclique. Also, $\langle u_1, p_1 \rangle$ and $\langle u_2, p_2 \rangle$ induce a biclique if another pair $\langle u_1, p_2 \rangle, \langle u_2, p_1 \rangle \in UP$ exists. Moreover, given $\omega_1, \omega_2 \in UP$, it can be easily verified that $\omega_1 \in biclique(\omega_2) \Leftrightarrow \omega_2 \in biclique(\omega_1)$ and $\omega_1 \in biclique(\omega_2) \Rightarrow \omega_1 \neq \omega_2$. Therefore, the undirected unipartite graph G induced from *UP* can be formally defined as:

$$G = \langle UP, \{\langle \omega_1, \omega_2 \rangle \in UP \times UP \,|\, \omega_1 \in biclique(\omega_2)\}\rangle. \quad (2)$$

Any clique partition of G corresponds to a biclique cover of B as well as to a possible solution for the role mining problem represented by the sets *USERS*, *UA*, and *PA* [6].

2.3. Jaccard and clustering coefficients

In this paper we extensively use two mathematical tools. The first one is represented by the *Jaccard coefficient* [18] that is a measure of the similarity between two sets. Given two sets S_1, S_2, the coefficient is defined as the size of the intersection divided by the size of their union:

$$J_{S_1 S_2} = |S_1 \cap S_2| \,/\, |S_1 \cup S_2|. \quad (3)$$

The Jaccard coefficient is widely used in statistic. However, to our knowledge, the only application to RBAC is given by Vaidya et al. [29], that offers a method to consider previously defined roles during the role mining process in order to minimize the "perturbation" introduced by new candidate roles. In Section 4.1 we will use the Jaccard index to measure the similarities among users of an access control configuration.

The other mathematical tool is the *clustering coefficient*. It was first introduced by Watts and Strogatz [31], in order to measure the cliquishness of a typical neighborhood. This coefficient became one of the central characteristics in complex network theory. In [24,25] the authors show that, in many real networks, the probability of having a relationship between two actors is much greater if the two actors have another mutual acquaintance, or several. An efficient clustering coefficient computation for large scale networks is described in [27].

The coefficient can be formally defined as follows. Let $G = \langle V, E \rangle$ be an undirected graph with a set of nodes V and a set of edges E. We indicate with $\delta(v)$ the number of *triangles* of v, formally:

$$\delta(v) = |\{\langle u,w \rangle \in E \,|\, \langle v,u \rangle \in E \wedge \langle v,w \rangle \in E\}|. \quad (4)$$

A path of length two for which v is the center node is called a *triple* of the vertex v. We indicate with $\tau(v)$ the number of triples of v, namely:

$$\tau(v) = |\{\langle u,w\rangle \in V \times V \,|\, \langle v,u\rangle \in E \wedge \langle v,w\rangle \in E\}|. \tag{5}$$

We now define the *clustering coefficient* of a graph G as:

$$C(G) = \frac{1}{|V|}\sum_{v\in V} c(v), \tag{6}$$

where

$$c(v) = \begin{cases} \dfrac{\delta(v)}{\tau(v)}, & \tau(v)\neq 0; \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

quantifies how close the vertex v and its neighbors are to being a clique. The quantity $c(v)$ is also referred to as the *local* clustering coefficient of v, while $C(G)$ is the average of all local clustering coefficients, and it is also referred to as the *global* clustering coefficient of G. Thus, $C(G)$ can be used to quantify "how well" a whole graph G is partitionable in cliques—in Section 4.2 we will further explain what "well" means in an access control scenario. Note that the provided definition differs from the classical definition of clustering coefficient. Indeed, its value is usually set to 0 when there are no triples. Our new definition is introduced because it is more suitable for role engineering applications, as will be clarified in Section 4.2, by contextualizing the cluster coefficient usage in the role mining process.

The clustering coefficient can also be written as:

$$C(G) = \frac{1}{|V|}\sum_{v\in V}\sum_{\pi\in\Pi_v}\frac{X(\pi)}{\tau(v)} + \frac{|\{v\in V \,|\, \tau(v)=0\}|}{|V|},$$

where $X:\Pi_v\rightarrow\{0,1\}$ is such that $X(\pi)$ is equal to 1 if there is an edge between the outer nodes of the triple π, and 0 otherwise. The first addendum is the minability of the subset of vertices that have $\tau(v)\neq 0$, while the second one corresponds to the minability of the set of vertices such that $\tau(v)=0$. Leveraging this definition of the clustering coefficient, $C(G)$ can be computed by considering each triple of the graph G, and then checking if it is a triangle or not.

3. Risk model

To better clarify the benefits introduced by the proposed approach, we first recall some risk management concepts. In particular, a typical risk management approach is made up of two key components: *risk analysis* (or *assessment*) and *risk control*. During risk analysis we identify potential risks and assess probabilities of negative events together with their consequences. With risk control we establish the tolerable level of risk for the organization, hence providing controls for failure prevention as well as actions to reduce the likelihood of a negative event—such an activity is usually referred to as *risk mitigation*.

Plugging the previous concepts in a RBAC environment, three essential components should be considered: users, roles, and permissions. Among them, particular attention must be taken on risk incurred by users. Indeed, the main threat in an access control scenario is to allow a user to execute an illegitimate operation over an object or a resource. A system which is only supposed to be used by authorized users must attempt to detect and exclude unauthorized ones. Accesses are therefore usually controlled by insisting on an authentication procedure to determine with some established degree of confidence the identity of the user, hence granting permissions authorized to that identity. RBAC mitigate the risk of unauthorized accesses by restricting user's permission to predefined role definitions. In a usual RBAC setting, users are assigned to roles which are then granted permissions to perform predefined tasks [15].

In this scenario, and assuming that it is not possible to by-pass the access control mechanism, the risk of illegitimate credentials is prevented by adopting a RBAC system. But, there is an important aspect that has not been considered so far: the role lifecycle. Roles are not static, but they follow the evolution of the organization: new users may join, existing users may leave or may change their job position, applications may be replaced with new ones, etc.. Hence, an important aspect to consider when evaluating the risks related to RBAC systems is the risk introduced by roles that are difficult to manage, mainly due to an unclear understanding of their meaning. Indeed, the more a role is intelligible and well designed, the less error prone it will be. A comprehensive risk management approach should consider these aspects starting from the creation of roles, that is the role mining phase. More specifically, we focus on the following risk-related aspects of a generic RBAC system:

- *Vulnerabilities*. They corresponds to roles that are not meaningful enough from the administrator's perspective, namely roles that are difficult to manage and to maintain.
- *Threats*. They are represented by errors and wrong administration actions, unintentionally committed while managing roles during their lifecycle.
- *Risks*. They correspond to allowing users to execute operations that are not permitted, or hampering their jobs by not granting required permissions. In both cases, the consequences could raise financial loss.

To evaluate such risks, in this paper we propose a general risk formula that involves multiple factors with different probabilities, namely:

$$Risk = \sum_{i=1}^{n} P_i \times C_i, \tag{8}$$

where P_i denotes the probability of each risk factor i, and C_i quantifies the consequences of these risk factors. In our model, risk factors are represented by homogeneous groups of users. Indeed, every user does not have the same degree of importance. For example, there could be users in charge of activities that are critical for the main business of the organization, while other users could be assigned to roles that have a marginal importance for the business. In general, we need to assign various degree of importance to each risk factor by taking the consequence of its execution into consideration. This process requires a thorough analysis of the organization. We assume that the impact evaluation is provided by experts. As for the probability of occurrence, we are able to propose two metrics that are suitable to evaluate the likelihood that an administration error is made when managing roles. In such a way, we evaluate the risk of an error in role management, and subsequently we are able to drive the definition of roles that mitigate this risk.

4. New metrics for role mining

In this section we describe two indices that can help role engineers to condition role mining in order to craft roles with business meaning and to downsize the problem complexity, hence reducing risk issues as well as required role mining effort. The first index is referred to as *similarity*, and its value is proportional to the number of permissions a given set of users share. The second one is *minability*, and it measures the complexity of selecting candidate roles given a set of user–permission assignments. Both indices provide a measure of how easy it is to analyze a given set of user–permission assignments through a bottom-up approach, but using different perspectives.

Indeed, roles may be classified in two categories, as described in [10, Ch. 5] and [23]:

- *Organizational or structural roles*, which depend on employee's position within a homogeneous group of users—for instance, an organization unit or all users that have the same job title. Common permissions are usually assigned to these kinds of roles, and each user typically has only one organizational role.
- *Functional roles*, which depend on the task that need to be performed in a particular position. Detailed permissions are usually assigned to this kind of roles. Functional roles are supposed to provide further access rights in addition to those being granted by organizational roles. Any number of functional roles can be assigned to a user.

Since the similarity index helps identify situations where all users share the majority of their permissions, its usage is most suitable when evaluating how easy identifying organizational roles is. Instead, the minability index indicates the level of complexity involved in identifying subsets of users that share the same permissions; thus, it is suitable to evaluate whether finding roles, both functional and organizational, is a simple task or not.

By leveraging the previous observations, it possible to use information resulting from a top-down approach in order to drive a bottom-up analysis. The key idea is to partition the data-set to analyze into smaller subsets according to some business information. For instance, a top-down analysis might identify groups of users that are homogeneous from an enterprise perspective. Then, user–permission assignments can be partitioned such that each subset contains only assignments related to users of the same group. For each subset, we calculate the minability and similarity indices, thus getting a prediction about how complex a subsequent role mining task on the subset will be. This decomposition process can be performed for each available business information, thus generating different data partitions; by selecting the one with the highest index values, we choose the partition that most simplifies the subsequent role mining analysis. Furthermore, each subset might be iteratively partitioned in even smaller subsets until we reach a given threshold for minability and similarity. The application of role mining algorithms on each subset will produce roles with more business meaning when compared to the outcome of the same algorithms applied on the whole data-set: since subsets are identified according to some business criteria, elicited roles will likely have a business meaning and the probability that administrators select a wrong role to assign to users will decrease, hence reducing the related risk.

In the following, we formally describe the minability and similarity indices, then in Section 5 we will introduce a methodology to apply them in practice within the proposed risk model.

4.1. Similarity

In order to introduce the similarity index, we leverage the Jaccard coefficient defined in Section 2.3 In particular, referring to Eq. (3), we provide the following definition:

Definition 1. (Similarity Between Two Users)

Given two users $u_1, u_2 \in USERS$, the *similarity index* between them is formally defined as:

$$s(u_1, u_2) = \frac{|perms(u_1) \cap perms(u_2)|}{|perms(u_1) \cup perms(u_2)|}. \tag{9}$$

The following observations are useful for the remainder of the paper:

- u_1 "contains" u_2 if $perms(u_1) \supset perms(u_2)$, namely permissions of u_1 are also possessed by u_2, but are not equal ($perms(u_1) \neq perms(u_2)$). In such a case, $s(u_1, u_2) \in (0,1)$.
- u_1 is "equivalent" to u_2 if $perms(u_1) = perms(u_2)$, namely u_1, u_2 share the same permission set. This condition matches with $s(u_1, u_2) = 1$.
- u_1 "overlaps" u_2 when $perms(u_1) \cap perms(u_2) \neq \emptyset$ but $perms(u_1) \not\supseteq perms(u_2)$ and $perms(u_2) \not\supseteq perms(u_1)$, namely u_1, u_2 share some permission but neither does u_1 contain u_2 nor does u_2 contain u_1. This means that $s(u_1, u_2) \in (0,1)$.
- u_1 is "not related" to u_2 if $perms(u_1) \cap perms(u_2) = \emptyset$, namely u_1, u_2 do not share any common permissions. This corresponds to $s(u_1, u_2) = 0$.

Definition 2. (Similarity Among a Set of Users)

Given a set of users *USERS*, the similarity index is the average similarity between all possible (unordered) user pairs. Formally,

$$S(USERS) = \begin{cases} \dfrac{1}{\binom{|USERS|}{2}} \displaystyle\sum_{u_1, u_2 \in USERS : u_1 \neq u_2} s(u_1, u_2), & |USERS| > 1; \\ 1, & \text{otherwise}. \end{cases} \tag{10}$$

Notice that Eqs. (9) and (10) can be extended to also consider other enterprise information. For instance, similarities can be evaluated over shared activities, involved organization units, etc.. We can define a similarity index for each kind of business data. In general, the most suitable similarity definition depends on specific organization needs and role engineering requirements. To ease exposition, in this paper the term "similarity" indicates only the percentage of permissions shared among users, according to the previous definitions.

4.2. Minability

The minability index measures how complex it is to identify and select the roles required to manage existing user–permission assignments. To do this, we will redefine the clustering coefficient (see Section 2.3) to be used with bipartite graphs that represent the user–permission assignments of an organization. In particular, given a user–permission assignment $\omega \in UP$, we define the function $triples : UP \to 2^{UP \times UP}$ as

$$triples(\omega) = \{\langle \omega_1, \omega_2 \rangle \in UP \times UP \mid \omega_1, \omega_2 \in biclique(\omega) \wedge \omega_1 \neq \omega_2\}, \tag{11}$$

namely the set of all possible pairs of elements in *UP* that both induce a biclique with ω. We also define the function $triangles : UP \to 2^{UP \times UP}$ as

$$triangles(\omega) = \{\langle \omega_1, \omega_2 \rangle \in triples(\omega) \mid \omega_1 \in biclique(\omega_2)\}, \tag{12}$$

namely the set of all possible pairs of elements in *UP* that both induce a biclique with ω, and that also induce a biclique with each other.

Definition 3. (Minability)

The *minability index* of an access control system configuration represented by the set *UP* is defined as

$$M(UP) = \frac{1}{|UP|} \sum_{\omega \in UP} m(\omega), \tag{13}$$

where

$$m(\omega) = \begin{cases} \dfrac{|triangles(\omega)|}{|triples(\omega)|}, & triples(\omega) \neq \emptyset; \\ 1, & \text{otherwise}. \end{cases} \tag{14}$$

The value of $m(\omega)$ is also referred to as the *local* minability index of ω, and it quantifies how close ω, together with all the edges which induce a biclique with it, are to being a biclique. Hence $M(UP)$, also referred to as the *global* minability index of *UP*, quantifies how well the bipartite graph, induced by the user–permission relations *UP*, is coverable with distinct bicliques. In other words, how easily a candidate role-set for the analyzed data can be identified—see below for a definition of "easy". Notice that, according to Eq. (14), when a user–permission assignment does not induce biclique with any other assignment, or it induces biclique with just one other assignment, its local minability is conventionally set to 1. This case is identified by $|triples(\omega)| = 0$. Note that when $triples(\omega) = 0$, the assignment ω induces a biclique with at most another user–permission assignment. Thus, the identification and selection of the unique role to manage ω is trivial.

Eq. (13) can be alternatively written as

$$M(UP) = \frac{1}{|UP|}\sum_{\omega\in UP}\ \sum_{\langle\omega_1,\omega_2\rangle\in triples(\omega)} \frac{Y(\omega,\omega_1,\omega_2)}{|triples(\omega)|} + \frac{|\{\omega\in UP\,|\,triples(\omega)=\varnothing\}|}{|UP|}, \tag{15}$$

where $Y: UP\times UP\times UP \to \{0,1\}$ returns 1 if their parameters induce a biclique, and 0 otherwise. Formally:

$$Y(\omega,\omega_1,\omega_2) = \begin{cases} 1, & \langle\omega_1,\omega_2\rangle \in triangles(\omega); \\ 0, & \text{otherwise}. \end{cases}$$

With the above formulation, the minability index $M(UP)$ can be computed by considering each tuple $\langle\omega,\omega_1,\omega_2\rangle \in UP\times UP\times UP$ such that $\omega_1,\omega_2 \in biclique(\omega)$, and checking whether the condition $\omega_1 \in biclique(\omega_2)$ holds true.

The following lemma defines a mapping between the clustering coefficient, as defined in Eq. (6), and the minability index:

Lemma 1. *If G is the unipartite graph constructed from UP according to Eq. (2), then $M(UP) = C(G)$, that is the minability of UP is equal to the clustering coefficient of G.*

Proof. It follows by construction of the graph G in Eq. (2), and definitions of $M(UP)$ in Eq. (13) and $C(G)$ in Eq. (6). □

Lemma 1 will be used in Sections 4.3 and 5.3 to offer a graph representation for the given examples. A relevant observation relates similarity with minability, and it is represented by the following lemma:

Lemma 2. *Given a set of users USERS that have been granted permissions in PERMS through the corresponding assignments UP, then $S(USERS) = 1 \Rightarrow M(UP) = 1$.*

Proof. When $S(USERS) = 1$ and $|USERS| = 1$, there is only one user that possesses all the permissions in *PERMS*. If $S(USERS) = 1$ and $|USERS| > 1$, then all users share the same permission set, namely $\forall u_1, u_2 \in USERS: perms(u_1) = perms(u_2)$. In both cases, $UP = USERS \times PERMS$. According to Eqs. (11) and (12), $\forall \omega \in UP: triples(\omega) = triangles(\omega)$. The proof immediately follows from the minability definition (Eq. (13)). □

The previous lemma states that the similarity index is tighter than the minability index. Indeed, a similarity index equal to 1 requires that all users share the same set of permissions, whereas minability can be equal to 1 even if the users do not share all the same permissions. Further, notice that the inverse of Lemma 2 does not hold. The example depicted in Fig. 1(a) is one possible case of $M(UP) = 1$ and $S(USERS) < 1$. In the following section we will offer more details on this example.

In the remainder of this section we introduce definitions and prove theorems that help to better understand the relationship between the minability index and the complexity of the role mining problem. In particular, the following definition is required to formalize all the subsequent considerations:

Definition 4. (Maximal Equivalent Roles (MERs))

Given a role $\overline{r} \in ROLES$, it is a *maximal equivalent role* (MER) if:

$$\nexists U \subseteq USERS, \nexists P \subseteq PERMS : U \times P \subseteq UP \wedge ass\_users\left(\overline{r}\right) \times ass\_perms\left(\overline{r}\right) \subset U \times P. \tag{16}$$

Informally, a MER is a role that is "representative" of all possible subsets of permissions shared by a given set of users [4]. The key observation which is made regarding a MER is that two permissions which always occur together among users should simultaneously belong to the same candidate roles. Without further business semantics of access control data, a bottom-up approach to role engineering cannot differentiate between a role made up of two permissions and two roles that contain individual permissions. Moreover, defining roles made up of as many permissions as possible likely minimizes the administration effort of the RBAC system by

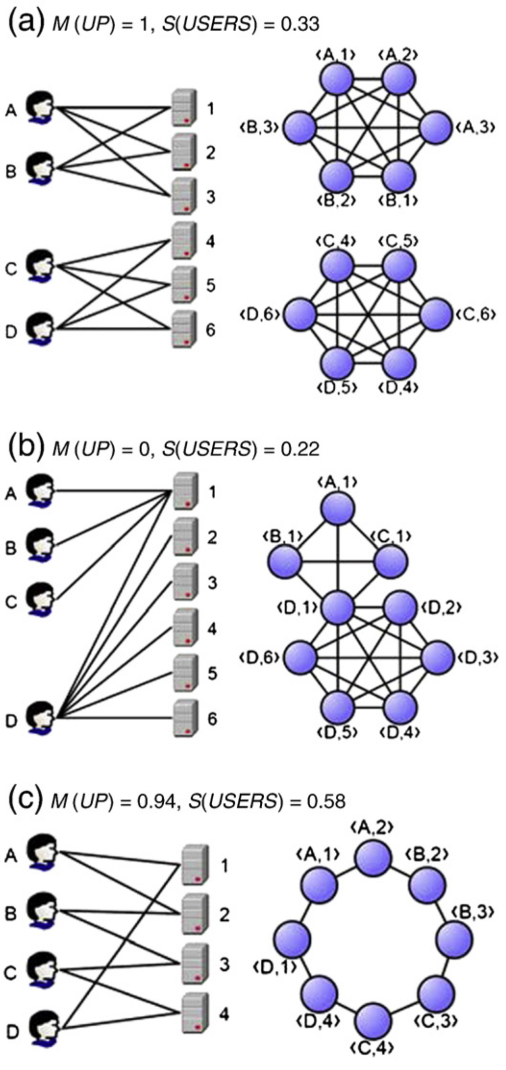


Fig. 1. Access control configurations as bipartite graphs and corresponding unipartite graphs.

reducing the number of required role–user assignments. MERs properties are further detailed in [4], which also proposes a variant of the *Apriori* algorithm to efficiently identify all possible MERs within *UP*.

The following theorem relates the minability index to the complexity of the role mining problem in terms of number of MERs:

Theorem 1. Let *ROLES* be the set of all possible MERs that can be derived from *UP*. Given a user–permission assignment $\omega \in UP$, let $MERS_\omega$ be the set of all possible MERs that "cover" the given user–permission assignment, namely $MERS_\omega = \{r \in ROLES \mid r \text{ is a MER} \wedge \omega \in ass_u sers(\bar{r}) \times ass_p erms(\bar{r})\}$. Then, the followings holds:

- $m(\omega) = 1 \Leftrightarrow |MERS_\omega| = 1$;
- $m(\omega) = 0 \Leftrightarrow |MERS_\omega| = |biclique(\omega)|$;
- $m(\omega) \in (0,1) \Leftrightarrow 1 < |MERS_\omega| < |biclique(\omega)|$.

Proof. First, we analyze the case $m(\omega) = 1$. Let $\bar{r}$ be a role made up of the users and permissions involved by the assignments ω and *biclique* (ω), formally $ass\_users(\bar{r}) = \{u \in USERS \mid \exists p \in PERMS, \langle u,p \rangle \in biclique(\omega) \cup \{\omega\}\}$ and $ass\_perms(\bar{r}) = \{p \in PERMS \mid \exists u \in USERS, \langle u,p \rangle \in biclique(\omega) \cup \{\omega\}\}$. We now demonstrate that r is a MER. Indeed, according to Eq. (1), $\forall \langle u_1,p_1 \rangle, \langle u_2,p_2 \rangle \in biclique(\omega) \cup \{\omega\} \Rightarrow \exists \langle u_1,p_2 \rangle, \langle u_2,p_1 \rangle \in UP$, namely both users u_1, u_2 have permissions p_1, p_2 been granted. According to Eq. (14), $m(\omega) = 1 \Rightarrow triples(\omega) = triangles(\omega)$, thus the previous consideration holds for every possible pair of user–permission relationships in $biclique(\omega) \cup \{\omega\}$. This means that $biclique(\omega) \cup \{\omega\} = ass\_users(\bar{r}) \times ass\_perms(\bar{r})$. Seeking a contradiction, if $\bar{r}$ were not a MER, two sets $U \subseteq USERS$ and $P \subseteq PERMS$ would exist such that $ass\_users(\bar{r}) \times ass\_perms(\bar{r}) \subset U \times P \subseteq UP$ (see Eq. (16)). Let $\omega = \langle u, p \rangle$. Yet, for each $\langle u', p' \rangle \in (U \times P) \setminus (ass\_users(\bar{r}) \times ass\_perms(\bar{r}))$ it can be easily shown that both the assignments $\langle u,p' \rangle, \langle u',p \rangle$ always exists in $U \times P$. Hence, according to Eq. (1), $\langle u',p' \rangle \in biclique(\omega)$, meaning that $(U \times P) \setminus (ass\_users(\bar{r}) \times ass\_perms(\bar{r})) = \emptyset$. Therefore, $\bar{r}$ is a MER. We now demonstrate that another MER that contains ω cannot exist. Indeed, if $\bar{r}'$ is a MER that contains ω (i.e., $\omega \in ass\_users(\bar{r}') \times ass\_perms(\bar{r}')$), for all $\omega' \in ass\_users(\bar{r}') \times ass\_perms(\bar{r}')$ $\{\omega\}$ it can be shown that $\omega' \in biclique(\omega)$. Hence, $\bar{r} = \bar{r}'$. Finally, having only one MER that contains ω implies that $m(\omega) = 1$. Let r be such a MER. Since it is the only MER, for each pair $\omega_1, \omega_2 \in ass\_users(\bar{r}) \times ass\_perms(\bar{r})$ such that $\omega_1 \neq \omega_2$ we have $\omega_1 \in biclique(\omega_2)$. Thus, $triples(\omega) = triangles(\omega)$, which corresponds to state that $m(\omega) = 1$. □

When $m(\omega) = 0$, we now demonstrate that it is possible to identify $|biclique(\omega)|$ distinct MERs made up of ω combined with each element of $biclique(\omega)$. Let $\omega = \langle u,p \rangle$. First, observe that such roles are distinct since $m(\omega) = 0 \Rightarrow triangles(\omega) = \emptyset$. We want to show that for each $\langle u_i,p_i \rangle \in biclique(\langle u,p \rangle)$, the role $\bar{r}_i$ such that $ass_u sers(\bar{r}_i) = \{u, u_i\}$ and $ass_p erms(\bar{r}_i) = \{p, p_i\}$ is a MER. Seeking a contradiction, if r_i were not a MER, two sets U $USERS$ and P $PERMS$ would exist such that $\{u,u_i\} \times \{p,p_i\} \subset U \times P \subseteq UP$ (see Eq. (16)). Let $\langle u',p' \rangle \in (U \times P) \setminus (\{u, u_i\} \times \{p,p_i\})$. It can be easily shown that $\langle u',p' \rangle \in biclique(\langle u_i,p_i \rangle)$, thus $triangles(\omega) \neq \emptyset$. But, according to Eq. (14), this means that $m(\omega) > 0$, which is a contradiction. Moreover, more than $|biclique(\omega)|$ distinct MERs that contain ω cannot exist. Indeed, let $n \in \mathbb{N} : n > |biclique(\omega)|$ be the number of the distinct MERs that contain ω. Let r_i indicate the ith MER, and let $\omega_i \in (ass\_users(\bar{r}_i) \times ass\_perms(\bar{r}_i))$ $\{\omega\}$. Thus, $\forall i \in 1 \dots n : \omega_i \in biclique(\omega)$, contradicting the inequality $biclique(\omega) < n$. We now prove that having $|biclique(\omega)|$ MERs implies that $m(\omega) = 0$. Let r_i indicate the ith MER, and let $\omega_i \in (ass\_users(\bar{r}_i) \times ass\_perms(\bar{r}_i))$ $\{\omega\}$. Since the roles are distinct, $\forall i,j \in 1 \dots |biclique(\omega)| : i \neq j$ we have that $\omega_i \notin biclique(\omega_j)$. Thus, $triangles(\omega) = \emptyset$ and, according to Eq. (14), $m(\omega) = 0$.

Finally, by excluding the previous two cases we merely have that $m(\omega) \in (0,1) \Leftrightarrow 1 < |MERS_\omega| < |biclique(\omega)|$.

The previous theorem allows us to make some consideration on the complexity of the role mining problem. Given a user–permission assignment ω, the higher its local minability is, the less the number of possible MERs to analyze is. The following subsection and Section 7 offer practical examples about this property.

4.3. Examples

We now show some examples which demonstrate how minability and similarity indices can actually provide role mining engineers with the expected complexity to find functional and/or organizational roles. In Fig. 1, three different and simple access control configurations are depicted. In each one, we have 4 users and 6 permissions, but with different user–permission assignments.

To better illustrate these indices, we also report the corresponding unipartite graphs constructed according to Eq. (2).

In Fig. 1(a) the minability index is 1. In this case, it is straightforward to verify that a clique cover of the unipartite graph is represented by $C_1 = \{\langle A,1 \rangle, \langle A,2 \rangle\ \langle A,3 \rangle, \langle B,1 \rangle, \langle B,2 \rangle, \langle B,3 \rangle\}$ and $C_2 = \{\langle C,4 \rangle, \langle C,5 \rangle, \langle C,6 \rangle, \langle D,4 \rangle, \langle D,5 \rangle, \langle D,6 \rangle\}$. In RBAC terms, C_1 and C_2 correspond to two maximal equivalent roles: the first one made up of permissions {1,2,3} and it is assigned with users {A,B}, the second one made up of permissions {4,5,6} and assigned with users {C,D}. As for the similarity index, $S(\{A,B,C,D\}) = 1/3$.

Fig. 1(b) shows another access control configuration, where the minability index is equal to 0. According to Th. 1, it represents the most ambiguous case. Indeed, in this example we have two possible MERs to manage each user–permission assignment (one is composed by one user and two permissions, the other one is made up of one permission and two users). Yet, without further business semantics of access control data, it is not clear which is the best choice. Another observation is that in Fig. 1(b) the similarity index is smaller than in Fig. 1(a). Indeed, since each permission is used by 2 users, it is impossible to define a role that has to be assigned to the majority of users.

Fig. 1(c) shows a slightly more complicated configuration, where the minability index is between 0 and 1, while the similarity is higher than in all previous cases. It is quite clear that the unipartite graph can be covered with two cliques (i.e., two MERs), and this suggests that the minability must be very close to 1. Indeed, we have an ambiguity only for the user–permission assignment $D,1$: it can belong to both the cliques $C_1 = \{\langle A,1 \rangle, \langle B,1 \rangle, \langle C,1 \rangle, \langle D,1 \rangle\}$ and $C_2 = \{\langle D,1 \rangle, \langle D,2 \rangle, \langle D,3 \rangle, \langle D,4 \rangle, \langle D,5 \rangle, \langle D,6 \rangle\}$. $M(UP) = 0.94$ is in line with the previous observation.

5. Applications of minability and similarity

As shown in the previous section, minability and similarity are estimates of the expected complexity to select roles within the role mining results. As a consequence, they are also metrics for the likelihood of making administration errors when managing roles throughout their lifecycle. For instance, given a group of users, when the minability index equals 1 for user–permission assignments involved in the group, according to Th. 1 there will be just one possible maximal equivalent role for managing those assignments. This means, for example, that new permissions introduced within the system will likely be assigned to all users of the group (via the single role) or to none of them, and new users that join such a group will likely be granted the same permissions of other users (namely, all the permissions contained within the single role). Moreover, the business meaning of the role is strictly related to business aspects that users within the group have in common. Therefore, the probability of making wrong access control decisions is low.

One possible application of the minability and similarity indices is to help data analysts guide a *divide-and-conquer* approach to role mining. Decomposing the role mining problem into smaller sub-problems is a best practice, especially when dealing with large datasets. Typical steps of a generic role mining process are [20]:

1. *Choice of information sources*. From the available data, a subset has to be selected which is most promising to yield suitable information for role creation.
2. *Data preparation*. The data is collected from the various locations, cleaned up from obvious or known as incorrect information, and transformed into a format in which it can then be processed by the data mining software.
3. *Exploration*. This phase is crucial for the whole role mining process. It will provide a “feeling” for the data contents and the expected role scheme. The results of this phase are suitable attribute sets for the unique representation of organizational and functional roles and suitable parameters for the role mining algorithms.
4. *Mining*. The role mining algorithm is performed.
5. *Role creation*. The outcome of the data mining run are used to derive candidate organizational and functional roles.
6. *Check, approval and implementation of resulting roles*. The resulting roles have to be checked for plausibility and correctness.
7. *User assignment*. The elicited roles are finally assigned to users.

In this scenario, leveraging minability and similarity allows for the identification of the business information that “best fits” with the access control data, namely the information that induces a decomposition which most simplifies the identification of a business meaning for roles elicited in the role mining phase. Both indices can be calculated in the exploration step and used to guide the subsequent role mining steps.

The reason why minability and similarity change after decomposing the problem can be analyzed in terms of the graph model described in Section 2.2. As a matter of fact, partitioning the set *UP* is equivalent to partitioning the unipartite graph *G* constructed according to Eq. (2) in subgraphs, since each element in *UP* corresponds to a node in *G*. Hence, partitioning *UP* means discarding all the relationships between user–permission assignments that are in two different subsets of the partition—they will no longer induce a biclique—, namely removing edges in *G* that connect distinct subgraphs. In other words, the partition “breaks” roles that spread across multiple subsets into more parts; that is, roles without a clear meaning according to the business information that induced the partition. However, an important problem arises: how to be sure that the roles selected to be broken down are less relevant from a business perspective. The following section explains how minability and similarity indices can practically be used in conjunction with a divide-and-conquer approach to role mining to elicit business roles and to mitigate enterprise risk.

5.1. Choosing the best decomposition

When we have several business information at our disposal (e.g., organization units, job titles, applications, etc.), we have to select the one that induces a partition for *UP*, which minimizes the risk for each subset and that simplifies the subsequent mining steps. The best partition can change depending on the organization needs. To guide the decomposition process, it is useful to have a metric that allows data analysts:

- To decide what business information most reduces the risk of a poor role definition and simplifies the subsequent mining steps.
- To predict whether splitting the problem into more sub-problems actually reduces the risk of having ill-defined roles. In particular, we can decide to iteratively decompose the data before executing the mining step, by applying a different decomposition at each iteration until given minability and similarity thresholds are reached for each subset.
- To verify that partitioning does not actually reduce the role mining complexity. If this is the case, access control information should thus be reviewed in order to improve their manageability.

In all previous cases, similarity and minability are a means to estimate the risk related to the data being analyzed. In fact, since both indices express the likelihood of making bad administration decisions due to the unclear meaning of roles, they can be used in conjunction with the risk formula (Eq. (8)) proposed in Section 3. Depending on the kind of roles that role engineers are looking for (organizational or functional), the similarity or the minability value can be combined with the importance of each subset to evaluate the risk of incurring a poor role design. In particular, the following indicators can support the partition selection problem:

Definition 5. (Similarity-Based Risk)

Let *USERS* be a set of users to analyze, and *PERMS*, *UP* be the corresponding permissions and assignments. The *similarity-based risk* of *USERS* is defined as:

$$Risk_S(USERS) = (1 - S(USERS)) \times C, \tag{17}$$

where *C* is the importance of the user group *USERS*, while $S(USERS)$ is the similarity value computed over the users belonging to *USERS*.

Definition 6. (Minability-Based Risk)

Let *UP* be a set of user–permission assignments between users in *USERS* and permissions in *PERMS*. The *minability-based risk* of *UP* is defined as:

$$Risk_M(UP) = (1 - M(UP)) \times C, \tag{18}$$

where *C* is the importance of the data represented by the assignment set *UP*, while $M(UP)$ is the similarity value computed over the user–permission assignments belonging to *UP*.

The previous definitions offer an estimate for the risk related to each subset. However, if the objective is to identify the best partition, we will compare the values obtained for similarity-based and/or minability-based risks on all subsets and choose the partition with the lowest “average” risk. Similarly, when we want to check if further decomposing the problem actually reduces the role mining complexity, we have to compare the “average” risk that we have with and without the decomposition. The following section shows a possible approach to summarize the risk related to a partition.

5.2. Conditioned indices

Instead of analyzing the risk values calculated on each subset of a given partition, in most case it is more advantageous to have a risk value that “summarizes” the simplification introduced by the partition. To this aim, we need to review all the abovementioned indices to *condition* them by the given partition. The conditioning concept will apply on both similarity and minability indices (we therefore speak of *conditioned similarity and minability*) and risk evaluation metrics (we therefore speak of *conditioned similarity- or minability-based risk*).

5.2.1. Conditioned similarity and similarity-based risk

Let $\Omega = \{\Omega_1, \ldots, \Omega_k\}$ be a *k*-partition of *UP* such that $\Omega_i \subseteq UP$ and $UP = \bigcup_{i=1}^{k} \Omega_i$. Each subset Ω_i induces a set of users Υ_i, such that $\Upsilon_i =$

$\{u \in USERS | \exists p \in PERMS, \langle u,p \rangle \in \Omega_i\}$. According to Eq. (10), we can define the similarity of Υ_i in the following way:

$$S(\Upsilon_i) = \begin{cases} \frac{1}{\binom{|\Upsilon_i|}{2}} \sum_{u_1,u_2 \in \Upsilon_i : u_1 \neq u_2} s_{\Omega_i}(u_1,u_2), |U| > 1; \\ 1, \qquad \text{otherwise}. \end{cases} \tag{19}$$

where $S_\Omega(u_1,u_2)$ is the similarity of the users u_1 and u_2 obtained by only considering the permissions that are involved in Ω_i. Eq. (19) can also be rewritten in the following way:

$$S(\Upsilon_i) = \frac{1}{\sigma_i + \binom{|\Upsilon_i|}{2}} \left(\sigma i + \sum_{\substack{u_1,u_2 \in \Upsilon_i : \\ u_1 \neq u_2}} s_{\Omega i}(u_1,u_2) \right),$$

where

$$\sigma_i = \begin{cases} 1, & |\Upsilon_i| = 1; \\ 0, & \text{otherwise}. \end{cases}$$

We can then offer the following definition:

Defintion 7. (Conditioned Similarity).

Given a partition $\Omega = \{\Omega_1,\ldots,\Omega_k\}$ for UP such that $UP = \bigcup_{i=1}^k \Omega_i$ and the induced sets of users $\Upsilon_i = \{u \in USERS | \exists p \in PERMS, \langle u,p \rangle \in \Omega_i\}$, we define the similarity index *conditioned by* Ω as

$$S_\Omega(USERS) = \frac{\sum_{i=1}^k S(\Upsilon_i)\left(\sigma_i + \binom{|\Upsilon_i|}{2}\right)}{\sum_{i=1}^k \left(\sigma_i + \binom{|\Upsilon_i|}{2}\right)} = \frac{\sum_{i=1}^k \sigma_i + \sum_{i=1}^k S(\Upsilon_i)\binom{|\Upsilon_i|}{2}}{\sum_{i=1}^k \sigma_i + \sum_{i=1}^k \binom{|\Upsilon_i|}{2}}. \tag{20}$$

Notice that Eq. (20) holds since $S(\Upsilon_i) = 1$ when $\sigma_i = 1$. Another important observation is that the conditioned index (Eq. (20)) is a sort of "modified" version of Eq. (10), where the pairs of users that belong to different subsets are discarded.

As for risk analysis, Def. 7 can be extended in order to take into account the importance of each subset. In particular, we provide the following definition:

Definition 8. (Conditioned Similarity-Based Risk)

Given a k-partition $\Omega = \{\Omega_1,\ldots,\Omega_k\}$ of UP and the induced sets of users $\Upsilon_i = \{u \in USERS | \exists p \in PERMS, \langle u,p \rangle \in \Omega_i\}$, we define the similarity-based risk *conditioned by* Ω as

$$Risk_{S_\Omega}(USERS, \Omega) = \frac{\sum_{i=1}^k (1 - S(\Upsilon_i))C_i\left(\sigma_i + \binom{|\Upsilon_i|}{2}\right)}{\sum_{i=1}^k \left(\sigma_i + \binom{|\Upsilon_i|}{2}\right)}, \tag{21}$$

where C_i is the importance of the user group Υ_i.

In particular, $Risk_{S_\Omega}(USERS, \Omega)$ is a weighted average of the risks related to each subset, where the weights are proportional to the subset cardinalities.

5.2.2. Conditioned minability and minability-based risk

Given a k-partition $\Omega = \{\Omega_1,\ldots,\Omega_k\}$ of UP, according to Eq. (13) the minability index of each subset Ω_i is

$$M(\Omega_i) = \frac{1}{|\Omega_i|} \sum_{\omega \in \Omega_i m_{\Omega_i}(\omega),}$$

where $m_{\Omega_i}(\omega)$ indicates the local minability of ω obtained considering only the user–permission assignments belonging to Ω_i. This leads to the following definition:

Definition 9. (Conditioned Minability)

Given a k-partition $\Omega = \{\Omega_1,\ldots,\Omega_k\}$ of UP the minability index *conditioned by* Ω is

$$M_\Omega(UP) = \frac{\sum_{i=1}^k M(\Omega_i)|\Omega_i|}{\sum_{i=1}^k |\Omega_i|} = \frac{1}{|UP|}\sum_{i=1}^k \sum_{\omega \in \Omega_i} m_{\Omega_i}(\omega) = \frac{1}{|UP|}\sum_{\omega \in UP} m_{\Omega_i}(\omega). \tag{22}$$

It is possible to note that the conditioned index (Eq. (22)) is similar to the basic minability index (Eq. (13)), except that relationships between user–permission assignments that belong to different subsets are no longer considered.

As for risk analysis, Def. 9 can be extended in order to take into account the importance of each subset. In particular, we provide the following definition:

Definition 10. (Conditioned Minability-Based Risk)

Given a k-partition $\Omega = \{\Omega_1,\ldots,\Omega_k\}$ of the set UP we define the minability-based risk *conditioned by* Ω as

$$Risk_{M_\Omega}(UP, \Omega) = \frac{\sum_{i=1}^k (1 - M(\Omega_i))C_i|\Omega_i|}{\sum_{i=1}^k |\Omega_i|}, \tag{23}$$

where C_i is the importance of the subset Ω_i.

In particular, $Risk_{M_\Omega}(UP,\Omega)$ is a weighted average of the risks related to each subset, where the weights are represented by the subset cardinalities.

5.3. Examples

In this subsection we show a simple application of our indices. Let us assume that the access control configuration to analyze is the one depicted in Fig. 2(a). The unipartite graph corresponding to the analyzed access control configuration is shown in Fig. 2(b). The values of the indices are reported in the caption. We now try to split the problem into several sub-problems by leveraging some available business information in order to check whether the minability and similarity values increase, and consequently the risk indices decrease. For this purpose, suppose that we have two different business information at our disposal: the organization unit the user belongs to, and the applications involved by the given permission set. This information is depicted in Fig. 2(a). In particular, the organization unit U_1 is composed of the users A, B, and C, while the organization units U_2 and U_3 are composed of the users D and E, respectively. As for the applications, A_x is composed of the permissions 1,2,3, and 4; A_y is composed of the permissions 5 and 6; while A_z is made up of permissions 7,8, and 9.

Given these pieces of information, we have to choose which one induces the partition that most simplifies the successive role mining steps. To ease exposition, we assume that all the subsets have the same importance. Fig. 2(c) shows the subsets generated by partitioning according to the organization units. Notice that both minability and similarity indices are equal to 1 for each subset, whereas the risk values are 0. Hence, the conditioned basic and risk indices equal 1 and 0, respectively. Fig. 2(d) shows the subsets generated by partitioning according to applications and the corresponding minability and similarity values for each subset. When comparing the conditioned indices of the two partitions, it can be easily seen that the

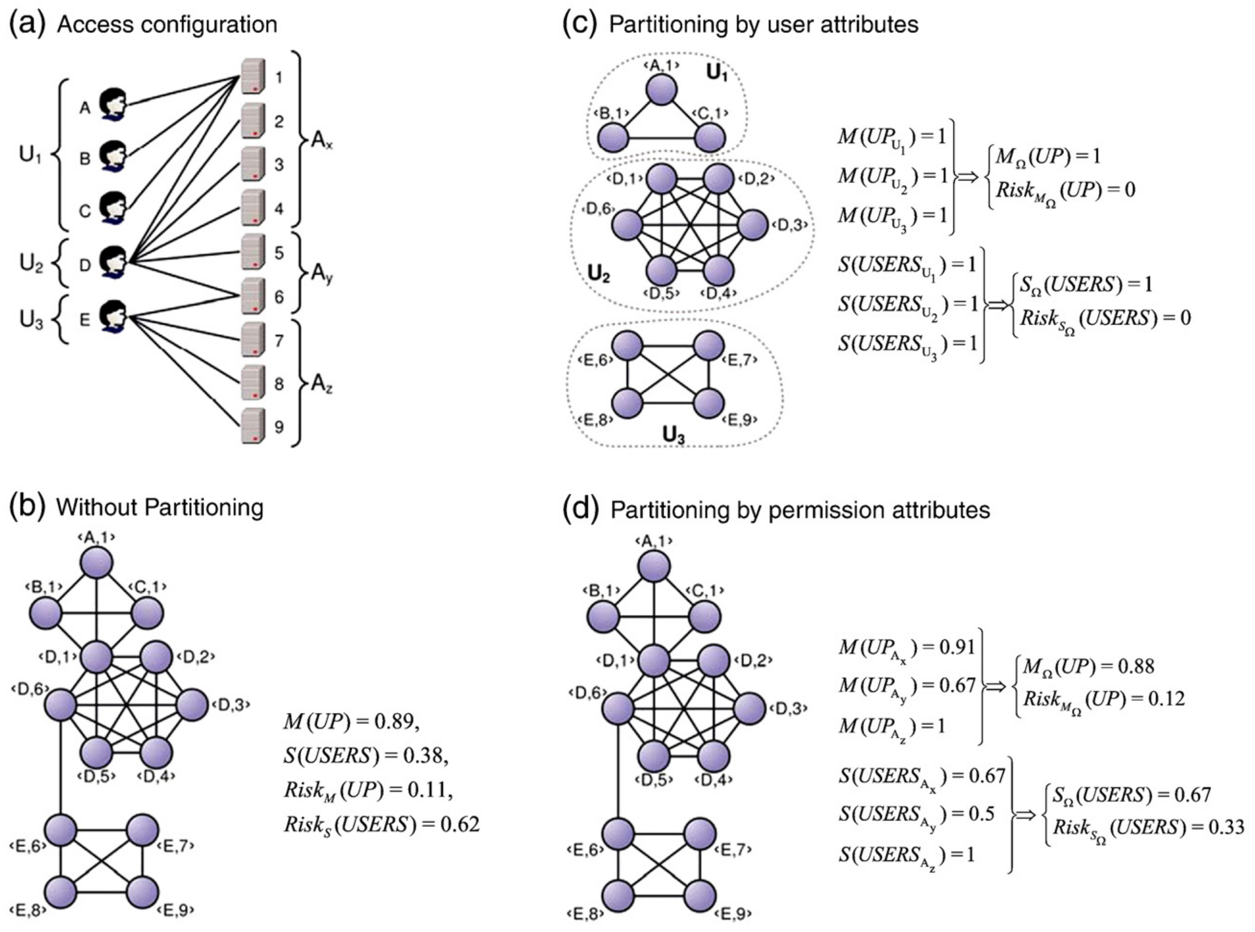


Fig. 2. A partitioning example.

organization-unit based partition is preferable to the applications-based partition. Indeed, the minability conditioned by applications is 0.88, which is even worse than the not-conditioned minability. Partitioning by organization units is also preferable when evaluating other conditioned indices.

6. Fast index approximation

In this section, we illustrate two algorithms to efficiently compute the similarity and minability indices introduced in Sections 4.1 and 4.2.

6.1. Approximating the similarity

Let us analyze the computation time required to determine the exact value of $S(USERS)$. In particular, according to its definition (Eq. (10)), this value can be calculated in $\mathcal{O}\left(|PERMS||USERS|^2\right)$ time. Indeed, $\mathcal{O}\left(|USERS|^2\right)$ time is required to identify all possible user pairs. For each pair $u_1,u_2 \in USERS$, the cardinality of both the intersection and the union of their granted permissions can be computed in $\mathcal{O}(|PERMS|)$. In particular, by scanning UP only once, we can build a hashtable of permissions that each user has been granted. Notice that $|UP| \leq |USERS||PERMS|$. Hence, checking if a permission is in $perms(u_1)$ requires $\mathcal{O}(1)$. This check should be done for every permission in $perms(u_2)$, thus requiring $\mathcal{O}(|PERMS|)$. Altogether, the similarity index can be calculated in $\mathcal{O}\left(|PERMS||USERS|^2\right)$.

Algorithm 1. Approximation of the similarity index.

```
1:  Procedure $\tilde{S}(USERS, k)$
2:      $\ell \leftarrow 0$
3:      if $|USERS| = 1$ then
4:          return 1
5:      else
6:          for $i = 1 \ldots k$ do
7:              Select $u_1, u_2 \in USERS : u_1 \neq u_2$ uniformly at random
8:              $\ell \leftarrow \ell + s(u_1, u_2)$
9:          end for
10:         return $\ell / k$
11:     end if
12: end procedure
```

To reduce the computation time, we propose the ε-approximated algorithm listed in Algorithm 1. The algorithm performs uniform sampling over all possible user pairs and then computes the average similarity among them. In particular, in each of the k sampling (Line 6), a user pair u_1,u_2 is randomly chosen (Line 7). Then, the variable ℓ is incremented by the similarity value of this pair (Line 8). In accordance with our definition, the returned result is ℓ/k. We now show that the algorithm is totally correct: it terminates in a finite time and provides a correct result. First, Algorithm 1 always terminates because its core is a finite loop. Then, the following theorem proves that the computed result is probabilistically correct:

Theorem 2. The value $\tilde{S}(USERS, k)$ computed by a run of Algorithm 1 satisfies:

$$\Pr\left(\left|\tilde{S}(USERS,k) - S(USERS)\right| \geq \varepsilon\right) \leq 2\exp\left(-2k\varepsilon^2\right).$$

Proof. If $|USERS| = 1$ the proof is immediate. Let us consider the case when $|USERS| > 1$. We will use the Hoeffding inequality [17] to prove this theorem. The cited inequality states that if $X_1 \ldots X_k$ are independent random variables such that $0 \le X_i \le 1$, then

$$\Pr\left(\left|\sum_{i=1}^{k} X_i - \mathbb{E}\left[\sum_{i=1}^{k} X_i\right]\right| \ge t\right) \le 2\exp\left(-\frac{2t^2}{k}\right), \quad (24)$$

where $\mathbb{E}[\cdot]$ indicates the expected value of a random variable. In our case, X_i indicates the similarity of a randomly chosen user pair. Eq. (24) can be rewritten as

$$\Pr\left(\left|\frac{1}{k}\sum_{i=1}^{k} X_i - \mathbb{E}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right]\right| \ge \varepsilon\right) \le 2\exp\left(-2k\varepsilon^2\right), \quad (25)$$

where $\varepsilon = t/k$. Notice that the value $\frac{1}{k}\sum_{i=1}^{k} X_i$ is exactly the output of Algorithm 1. Hence, in order to prove that the algorithm gives an approximation of $S(USERS)$, we have to prove that $\mathbb{E}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right]$ is equal to $S(USERS)$. Because of the linearity of the expectation, the following equation holds:

$$\mathbb{E}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right] = \frac{1}{k}\sum_{i=1}^{k} \mathbb{E}[X_i]. \quad (26)$$

Since the user pair used to calculate X_i is picked uniformly at random, the corresponding similarity value is produced with a probability of $1/\binom{|USERS|}{2}$—that is, one out of all the possible (unordered) pairs. Thus, the expected value of X_i is

$$\forall i \in 1 \ldots k, \quad \mathbb{E}[X_i] = \sum_{\substack{u_1, u_2 \in USER: \\ u_1 \ne u_2}} \frac{s(u_1, u_2)}{\binom{|USERS|}{2}}.$$

The previous equation is the definition of $S(USERS)$ as in Eq. (10) when $|USERS| > 1$, completing the proof. □

For practical applications of Algorithm 1, it is possible to calculate the number of loops needed to obtain an expected error that is less than ε with a probability greater than p. The following is an application of Th. 2:

$$k > -\frac{1}{2\varepsilon^2}\ln\left(\frac{1-p}{2}\right). \quad (27)$$

For instance, if we want an error $\varepsilon < 0.05$ with probability greater than 98.6%, it is enough to choose $k \ge 993$.

Finally, we shall demonstrate that the computational complexity of Algorithm 1 is $\mathcal{O}(|UP||PERMS|)$. Indeed, according to the observation made at the beginning of this section, we can build a hashtable of permissions possessed by users in $\mathcal{O}(UP)$. The loop in Line 6 is repeated k times, and we reasonably assume that $k \in \mathcal{O}(|UP|)$. Computing the similarity of two users in each loop requires $\mathcal{O}(|PERMS|)$ thanks to the hashtable. Therefore, the total complexity is $\mathcal{O}(|UP||PERMS|)$, that is advantageous when compared to the exact similarity calculation if the number of users is greater than the number of permissions and, most of all, when the user set is large.

6.2. Approximating the minability

Here we will show that the computational complexity of calculating $M(UP)$ is $\mathcal{O}\left(|UP|^3\right)$. In the case of a large-size organization, computation may be unfeasible since UP can count hundreds of thousands of user–permission assignments. For this reason, we propose an approximation algorithm for $M(UP)$ that has a computational complexity of $\mathcal{O}(k|UP|)$.

First, let us consider the complexity of computing the exact value of $M(UP)$. In Eq. (15), the first sum is over all the user–permission assignments $\omega \in UP$, while the second one is over all the triples $\langle\omega_1, \omega_2\rangle \in triples(\omega)$—that, for a given ω, are $(|UP|-1)(|UP|-2)$ in the worst case. Each addendum of the sum corresponds to checking whether the selected triple is also a triangle. It is a triangle if the two outer nodes $\omega_1 = \langle u, p\rangle$ and $\omega_2 = \langle u', p'\rangle$ of the selected triple induce a biclique. This occurs if $u = u'$, or $p = p'$, or other two edges $\omega_3 = \langle u, p'\rangle$ and $\omega_4 = \langle u', p\rangle$ exist in UP. It is possible to check if $u = u'$ or $p = p'$ in a constant time. Instead, the search for the pair ω_3, ω_4 can be executed in $\mathcal{O}(1)$ after having built a hashtable of all possible user–permission assignments in $\mathcal{O}(|UP|)$. The total computational cost is thus $\mathcal{O}\left(|UP|^3\right)$.

Algorithm 2. Approximation of the minability index.

```
1:  Procedure $\tilde{M}(UP, k)$
2:      $\ell \leftarrow 0$
3:      for $i = 1 \ldots k$ do
4:          Select $\omega \in UP$ uniformly at random
5:          if $triples(\omega) \ne \emptyset$ then
6:              Select $\langle\omega_1, \omega_2\rangle \in triples(\omega)$ uniformly at random
7:              if $\omega_1 \in biclique(\omega_2)$ then
8:                  $\ell \leftarrow \ell + 1$
9:              end if
10:         else
11:             $\ell \leftarrow \ell + 1$
12:         end if
13:     end for
14:     return $\ell / k$
15: end procedure
```

To reduce the computation time, we propose the ε-approximated algorithm listed in Algorithm 2, that is inspired by [27] but adapted to the bipartite graph case. In each of the k steps, a user–permission assignment ω is randomly chosen (Line 4). Then, two random user–permission assignments among those that induce a biclique together with ω (if any) are selected (Line 6). If these two user–permission assignments induce a biclique, the counter ℓ is incremented by 1 since we have found a triple that is also a triangle (Line 7). The ratio of the number of found triangles ℓ to the number of sampled triples k (Line 14) represents the approximated minability value.

In the following, we show that the algorithm terminates and returns a correct result. First, notice that the core of Algorithm 2 is a finite loop, thus it always outputs a result in a finite amount of time. The following theorem proves that this answer is probabilistically correct:

Theorem 3. The value $\tilde{M}(UP, k)$ computed by a run of Algorithm 2 satisfies:

$$Pr\left(\left|\tilde{M}(UP, k) - M(UP)\right| \ge \varepsilon\right) \le 2\exp\left(-2k\varepsilon^2\right).$$

Proof. We follow the same proof schema of Th. 2. Let $X_1 \ldots X_k$ be independent random variables, where $X_i = 1$ if, for a randomly selected tuple $\langle\omega, \omega_1, \omega_2\rangle$ $UP \times UP \times UP$ such that $\langle\omega_1, \omega_2\rangle \in triples(\omega)$, either $\omega_1 \in biclique(\omega_2)$ or $triples(\omega) = \emptyset$. Otherwise, $X_i = 0$. In this case, Eq. (25) still holds and, in particular, $\frac{1}{k}\sum_{i=1}^{k} X_i$ is exactly the output of Algorithm 2. Hence, in order to prove that the algorithm gives an approximation of $M(UP)$, we have to prove that $\mathbb{E}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right]$ is equal to $M(UP)$. Because of the linearity of the expectation, Eq. (26) still holds. To calculate X_i we first pick a user–permission relationship uniformly at random, then we pick two user–permission relationships that make up a triple. Thus, the corresponding minability value is produced with a probability of $1/(|UP||triples(\omega)|)$. Consequently, the expected value of X_i is

$$\forall i \in 1 \ldots k, \mathbb{E}[X_i] = \sum_{\omega \in UP} \sum_{\langle \omega_1, \omega_2 \rangle \in triples(\omega)} \frac{Y(\omega, \omega_1, \omega_2)}{|UP||triples(\omega)|} + \sum_{\omega \in UP: triples(\omega) = \varnothing} \frac{1}{|UP|}.$$

The previous equation is equivalent to the definition of $M(UP)$ as in Eq. (15), completing the proof. □

In the same way as the similarity index, it is possible to calculate the number of times it takes the loop in Algorithm 2 to obtain an expected error which is less than ε with a probability greater than p. By analyzing Th. 3 it can be seen that the same result (Eq. (27)) holds for this case.

As for computational complexity, we will now show that Algorithm 2 requires a time $\mathcal{O}(k|UP|)$ to run. The loop in Line 3 is repeated k times. In each loop, a random user–permission relationship $\omega \in UP$ can be selected in constant time (Line 4). Let us consider $\omega = \langle u, p \rangle$. In order to randomly select a pair $\langle \omega_1, \omega_2 \rangle$ that belongs to *triples* (ω), we have to calculate the set *biclique*(ω), then every possible pair of this set is in *triples*(ω) (Line 5). Eq. (1) states that $biclique(\omega) = \{\langle u', p' \rangle \in UP | \langle u, p' \rangle, \langle u', p \rangle \in UP \wedge \langle u, p \rangle \neq \langle u', p' \rangle\}$. The number of elements of *biclique*(ω) is at most $|UP|-1$, and each element can be found in $\mathcal{O}(1)$ after having built a hashtable of all possible user–permission assignments in $\mathcal{O}(|UP|)$. Then, the computational cost incurred to identify a biclique is at most $\mathcal{O}(|UP|)$. Line 7 can be executed in $\mathcal{O}(1)$ since it represents a search in the hashtable to verify the conditions in Eq. (1). Hence, the computational complexity of Algorithm 2 is $\mathcal{O}(k|UP|)$, which greatly improves over the time required to calculate the exact value $M(UP)$. This computational improvement is traded-off with a slight (tunable) decrease in the precision of the computed value $M(UP, k)$.

6.3. Approximation of conditioned indices

We now demonstrate that the amount of approximation introduced by the proposed randomized algorithms, when applied to the calculation of conditioned indices, is comparable to the approximation of the non-conditioned indices.

Theorem 4. Let $S_\Omega(USERS)$, $Risk_{S_\Omega}(USERS, \Omega)$, $M_\Omega(UP)$, and $Risk_{M_\Omega}(UP, \Omega)$ be the exact indices conditioned by a given partition Ω according to definitions 7, 8, 9, and 10, respectively. Let $\tilde{S}_\Omega(USERS, k)$, $\widetilde{Risk}_{S_\Omega}(USERS, \Omega, k)$, $\tilde{M}_\Omega(UP, k)$, and $\widetilde{Risk}_{M_\Omega}(UP, \Omega, k)$ be the corresponding approximated values computed by adopting Algorithm 1 and Algorithm 2 for each subset $\Omega_i \in \Omega$. Then:

$$\Pr\left(\left|\tilde{S}_\Omega(USERS, k) - S_\Omega(USERS)\right| \geq \varepsilon\right) \leq 2\exp\left(-2k\varepsilon^2\right)$$

$$\Pr\left(\left|\widetilde{Risk}_{S_\Omega}(USERS, \Omega, k) - Risk_{S_\Omega}(USERS, \Omega)\right| \geq \varepsilon\right) \leq 2\exp\left(-2k\varepsilon^2\right)$$

$$\Pr\left(\left|\tilde{M}_\Omega(UP, k) - M_\Omega(UP)\right| \geq \varepsilon\right) \leq 2\exp\left(-2k\varepsilon^2\right)$$

$$\Pr\left(\left|\widetilde{Risk}_{M_\Omega}(UP, \Omega, k) - Risk_{M_\Omega}(UP, \Omega)\right| \geq \varepsilon\right) \leq 2\exp\left(-2k\varepsilon^2\right).$$

Proof. We first demonstrate that the theorem holds true for the approximation introduced by the given algorithms for the conditioned minability. Let ε_i be the approximation for each subset Ω_i, namely:

$$\tilde{M}(\Omega_i, k) = M(\Omega_i) + \varepsilon_i$$

The approximated conditioned value will thus be:

$$\tilde{M}_\Omega(UP, k) = \frac{\sum \tilde{M}(\Omega_i, k)|\Omega_i|}{\sum |\Omega_i|} = M_\Omega(UP) + \frac{\sum \varepsilon_i |\Omega_i|}{\sum |\Omega_i|} \quad (28)$$

According to Th. 3, $|\varepsilon_i| \geq \varepsilon$ with probability less than $2\exp(-2k\varepsilon^2)$, hence $\left|\frac{\sum \varepsilon_i |\Omega_i|}{\sum |\Omega_i|}\right| \geq \varepsilon$ with probability less than $2\exp(-2k\varepsilon^2)$. Put another way, $\Pr\left(\left|\tilde{M}_\Omega(UP, k) - M_\Omega(UP)\right| \geq \varepsilon\right) \leq 2\exp(-2k\varepsilon^2)$. Thus completing the proof for the approximated conditioned minability. The proof for other conditioned indices can simply be obtained by replacing Eq. (28) with the corresponding index definitions. □

7. Results and discussion

To demonstrate the usefulness of the proposed indices, we show how they have been applied to a real case. Our case study has been carried out on a large private organization. We examined a representative organization branch that contained 1363 users with 5319 granted permissions, resulting in a total of 84201 user–permission assignments. To apply our approach we used two information sources: the organization unit (OU) chart, and a categorization of the users based on their job titles. In order to protect organization privacy, all names reported in this paper for organization units and job titles are slightly different from the original ones. We calculated all the indices described in the previous sections by adopting Algorithm 1 and Algorithm 2 with $k = 5000$, hence obtaining an error of less than 0.02 with a probability higher than 96%.

The remainder of this section is organized as follows. In Section 7.1 we will show two examples that have different values for minability and similarity, thus making it possible to better understand the meaning of having high or low values associated to these indices. In turn, in Section 7.2 we will apply our methodology in order to select the best available top-down information to decompose the problem. To further demonstrate the reliability of the methodology, we borrow from biology the methodology of introducing a *control* test. That is, we try to categorize users according to the first character of their surname. Since this categorization does not reflect any access control logic, we will analytically show that—as expected—it never helps the mining phase. Finally, in Section 7.3 we will use the proposed methodology in conjunction with the organizational unit chart to "drill-down" into smaller role mining problems.

7.1. High and low values of minability and similarity

Fig. 3 shows the user–permission assignments for two distinct sets of users that belong to two chosen branches of the analyzed organization. The two OUs are comparable in terms of number of users, permissions, and user–permission assignments: Fig. 3(a) is related to 54 users who possess 285 permissions through 2379 user–permission assignments; Fig. 3(b) represents 48 users who possess 299 permissions through 2081 user–permission assignments.

Assignments are depicted in a matrix form, where each row represents a user, each column represents a permission, and a black cell indicates a user with a given permission granted. By using the role mining algorithm described in [4], we computed all possible maximal equivalent roles. Then, rows and columns have been sorted so that roles with the largest number of users and permissions appear as "big" areas of contiguous painted cells.

Fig. 3(a) is an example of high values for minability (0.84) and similarity (0.43). It visually demonstrates how, in this case, it is easy to identify candidate roles—few groups of contiguous cells that cover most of the assignments can be easily identified via a visual inspection. The role identification task clearly requires more effort

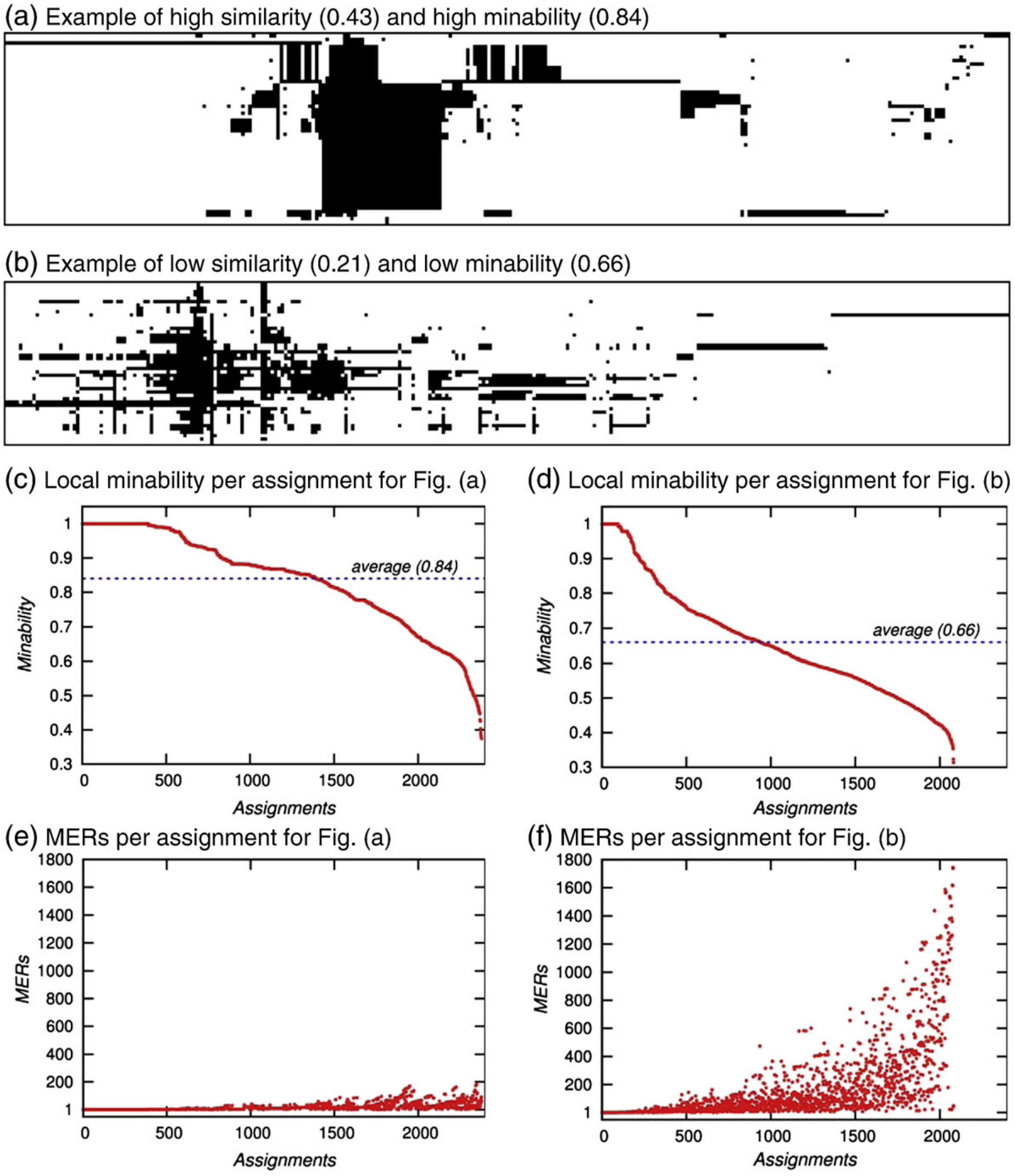


Fig. 3. Examples of different values for similarity and minability. Figures (a) and (b) depict user–permission assignments in a matrix form, where each black cell indicates a user (row) that has a certain permission (column) been granted. Figures (e) and (f) show the number of MERs which cover each user–permission assignment, sorted by the descending local minability values reported in (c) and (d).

in Fig. 3(b), in line with lower values for minability (0.66) and similarity (0.21). Indeed, it is impossible to define an organizational role composed of as many users and permissions as in the previous case, and it is harder to identify roles in general. This intuition is also supported by Fig. 3(e) and (f). In these pictures we show the number of possible MERs that can be used to manage each user–permission assignment of Fig. 3(a) and (b), respectively. Assignments are sorted by descending local minabilities, and the corresponding minability values are reported in Fig. 3(c) and (d). In the first case, the number of assignments with a local minability close to 1 is higher than in the second case. This is reflected by the number of MERs that cover each user–permission assignment, that is lower in the first case. Put another way, the ambiguity of selecting the role to manage each user–permission assignment is lower in the first example. This is in line with Th. 1, which states that when the local minability of an assignment is equal to 1, there is only one MER to choose. The more the minability is far from 1, the more the number of MERs that can be used to manage that assignment increases, indicating that the identification of the "best" role-set requires more effort and, consequently, it is more error prone.

7.2. Selection of the best business information

In this section we summarize how we have implemented our divide-and-conquer approach. As anticipated before, we had at our disposal two top-down pieces of information—OU and job titles—, and we wanted to choose the one that mostly simplifies the subsequent mining steps. As a *control*, we also introduced a third "artificial"

Table 1
Further decomposition of the sample organization branch.

Organization Unit	Index Type	Similarity	Minability	Similarity-Based Risk	Minability-Based Risk	Roles	msec
Operations							
Manufacturing							
	Not Conditioned	0.08	0.68	4.59	1.58	10,001	242
	Conditioned by Alphabetical Groups	0.08	0.74	7.30	1.70	4,422	134
	Conditioned by Organization Units	0.10	0.78	4.40	0.97	4,424	125
	Conditioned by Job Titles	0.29	0.89	3.23	0.53	918	59
Product Development							
	Not Conditioned	0.20	0.82	4.01	0.92	4,007	150
	Conditioned by Alphabetical Groups	0.20	0.84	6.24	1.11	2,511	73
	Conditioned by Organization Units	0.37	0.86	4.83	1.05	2,080	76
	Conditioned by Job Titles	0.38	0.90	5.63	0.64	818	36
Material Management							
	Not Conditioned	0.28	0.76	0.72	0.24	36,620	276
	Conditioned by Alphabetical Groups	0.28	0.80	5.58	1.28	3,504	48
	Conditioned by Organization Units	0.33	0.85	0.69	0.17	1,224	25
	Conditioned by Job Titles	0.28	0.82	0.72	0.21	21,614	105
Sales							
	Not Conditioned	0.07	0.63	4.64	1.85	61,933	471
	Conditioned by Alphabetical Groups	0.08	0.72	8.15	2.23	11,659	81
	Conditioned by Organization Units	0.11	0.67	1.63	1.39	50,314	301
	Conditioned by Job Titles	0.11	0.80	4.21	0.73	1,757	30
Quality							
	Not Conditioned	0.08	0.89	4.61	0.57	40	2
	Conditioned by Alphabetical Groups	0.08	0.94	8.35	0.58	36	1
	Conditioned by Organization Units	0.12	0.94	5.92	0.45	24	2
	Conditioned by Job Titles	0.21	0.96	3.61	0.22	25	1
Logistics							
	Not Conditioned	0.24	0.71	3.78	1.43	1,677	19
	Conditioned by Alphabetical Groups	0.24	0.80	6.47	1.60	642	13
	Conditioned by Organization Units	0.33	0.80	3.35	0.87	854	16
	Conditioned by Job Titles	0.64	0.83	0.59	0.45	357	12

information without any relation to the business—the first letter of user's surname. We generated three groups of users: A–G, H–P, and Q–Z. Obviously, we did not expect that this information would help the identification of roles. Indeed, experimental results confirmed our expectations, as shown later on.

The job title information was only available inside OU branches at the second level of the OU tree. Therefore, we first decomposed the problem by using the first OU level. Table 1 sums up the results for one of the first level OUs, namely the branch Operations. As required by Def. 8 and Def. 10, for both OUs and job titles we estimated the impact of harmful administration actions for each potential group of users—due to the large number of involved job titles, in Table 2 we only report the impact classification for OUs. Each group of users had been classified as “Low”, “Medium”, or “High” impact. Adopting a three-point scale made it easier to reach consensus among administrators. The values assigned to those impact classes were conventionally set by administrators to 1, 5, and 10, respectively.

For each index, the best values among all available partitions is highlighted in gray in Table 1. First, notice that alphabetical groups are never preferred, since they do not capture any pattern or commonality among users within access control data. For the other pieces of information, different cases can be identified:

- In ‘Manufacturing’, partitioning by job titles is the best choice according to all indices. This means that job title is a good user's attribute to use when defining administration rules for the assignment of roles with users belonging to ‘Manufacturing’.
- Even though the job title concept is closer to the “role” concept, partitioning by job titles is not always the best choice. Material Management shows a case where the best partition is based on OUs. In this case, this is justified by the fact that the majority of the users have the same job title. Hence, partitioning does not actually improve the mining complexity.
- The unit Sales shows a configuration where the minability and similarity indices suggest to partition by job title, but the risk indices promote the OU information. This happens because the unit Logistics contains many users that have a medium impact, and such users are not as similar among them as those having job titles with medium impact.
- Partitioning is *not* always advantageous. For instance, all users within the unit ‘Product Development’ have some commonalities in their permissions that will be lost when decomposing—as a matter of fact, users within Marketing have a medium impact and are not similar among them. Thus, if the role engineering objective is to find organizational roles for ‘Product Development’, it is better to analyze the unit as a whole.
- As for the mining complexity, the number of maximal roles elicited by the role mining algorithm [4] is in line with the minability and similarity indices. Few roles also means less elaboration time, thus resulting in faster algorithm runs.

Table 2
Further decomposition of the sample organization branch.

Organization Unit	Users	Permissions	User-Perms Assignments	Impact	Similarity	Minability	Similarity Based Risk	Minability-Based Risk	Roles	msec
Operations	**946**	**3,647**	**56,905**	**Medium**	**0.07**	**0.65**	**4.63**	**1.73**	**219,086**	**3 637**
Manufacturing	**379**	**1,810**	**20,400**	**Medium**	**0.08**	**0.68**	**4.59**	1.58	**10,001**	**242**
Parent	1	64	64	Low	1.00	1.00	0.00	0.00	1	0
Technology	49	323	2,342	Low	0.35	0.92	0.65	0.08	127	6
Control	26	577	2,396	Low	0.30	0.81	0.70	0.19	254	7
Test & Quality	8	226	301	Low	0.09	0.92	0.91	0.08	17	1
Production	288	1,452	15,226	Medium	0.09	0.75	0.54	1.26	4 013	111
Plants	7	39	71	Low	0.12	0.83	0.88	0.17	12	0
Product Development	**319**	**1,341**	**14,222**	**Medium**	**0.20**	0.82	**4.01**	**0.92**	**4 007**	**150**
Parent	2	32	33	Low	0.03	0.98	0.97	0.02	3	0
Engineering	77	559	2,898	Medium	0.36	0.82	3.22	0.90	564	16
Design #1	88	361	3,275	Medium	0.41	0.84	2.94	0.80	10	1
Design #2	121	739	6,965	High	0.87	0.87	6.48	1.34	232	10
Design #3	6	115	214	Medium	0.14	0.93	4.28	0.36	1 205	47
Marketing	17	404	663	Medium	0.08	0.91	4.58	0.45	57	2
Innovation	8	92	174	Medium	0.23	0.97	3.85	0.16	9	0
Material Management	**58**	**1,038**	**8,670**	Low	**0.28**	**0.76**	0.72	**0.24**	**36,620**	**276**
Parent	3	286	368	Low	0.17	0.95	0.83	0.05	6	1
Purchase Dept #1	23	549	3,043	Low	0.27	0.80	0.73	0.20	745	10
Purchase Dept #2	13	407	2,136	Low	0.49	0.89	0.51	0.11	382	6
Purchase Dept #3	7	406	1,054	Low	0.29	0.77	0.71	0.23	56	3
Purchase Dept #4	5	311	972	Low	0.69	0.91	0.31	0.09	15	2
Saving Control	3	203	303	Medium	0.32	0.86	0.40	0.70	7	1
Analysis & Reproting	4	376	794	Low	0.35	0.87	0.65	0.13	13	2
Sales	**100**	**1,531**	**9,483**	**Medium**	**0.07**	**0.63**	**4.64**	**1.85**	**61,933**	**471**
Parent	1	68	68	Low	1.00	1.00	0.00	0.00	1	0
Logistics	36	1,142	6,836	Medium	0.24	0.63	3.78	1.84	49,256	279
Support	63	795	2,579	Low	0.07	0.76	0.93	0.24	1 057	22
Quality	**16**	**272**	**697**	**Medium**	**0.08**	**0.89**	**4.61**	**0.57**	**40**	**2**
Parent	1	20	20	Low	1.00	1.00	0.00	0.00	1	0
Certification	2	40	46	Low	0.15	0.92	0.85	0.08	3	0
Audit	6	188	352	High	0.20	0.94	8.00	0.60	7	1
Quality Center	7	151	279	Medium	0.07	0.93	4.67	0.36	13	1
Logistics	**73**	**1,078**	**3,432**	**Medium**	**0.24**	**0.71**	**3.78**	1.43	**1 677**	**19**
Parent	0	0	0	Low	-	-	1.00	1.00	0	0
Methodologies	2	350	549	Low	0.57	0.92	0.43	0.08	3	1
Planning	7	311	682	Low	0.41	0.88	0.59	0.12	38	2
Distribution #1	62	464	1,780	Medium	0.32	0.70	3.38	1.52	810	12
Distribution #2	2	351	241	Medium	0.20	0.93	4.00	0.37	3	1

The previous examples also demonstrate that, in general, the choice of the best index to use (similarity, minability, similarity-based risk index, or minability-based risk index) depends on the main objective of the role engineering task.

7.3. Drill down

We now show an application of the proposed methodology when hierarchical information is available. Suppose that the only available top-down information is the organizational unit chart. Table 2 shows index values obtained by iteratively applying a decomposition based on OUs for the unit Operations. First, notice that partitioning users according to the second level of the OU tree raised the values of both indices for most OUs. For instance, 'Product Development', which holds approximately one third of the branch Operations, has a minability of 0.82 and a similarity of 0.20. Conversely, 'Manufacturing' still has low values for those indices. This means that it would be easier to find optimal organizational and functional roles for 'Product Development' rather than for 'Manufacturing'. Moreover, the average increase of minability is reflected by a lower number of possible MERs, dropped down from 219,086 to 114,278.

Another observation is that partitioning always reduces the number of users, permissions, and user–permission assignments to analyze, but the minability and the similarity values do not rise proportionally. For example, Logistics has a minability of 0.71, but his child 'Distribution #1' has a minability of 0.70, indicating that the "mess" of Logistics is likely concentrated in 'Distribution #1', as confirmed by the number of MERs. Further, 'Product Development' has much more users than Logistics, but it also has a higher minability. Moreover, high values for similarity imply high minability as well, but the inverse does not hold. If similarity is close to 1, then all users possess the same permissions; thus, minability is also close to 1. The opposite is false. For example, 'Distribution #2' shows a high minability (0.93) and a low similarity (0.20).

There are other examples of different trends for minability and similarity when compared to the number of users or permissions. For instance, let us consider 'Marketing' and 'Purchase Dept #2' which have similar number of users and permissions. In the first case, we have 0.91 for minability and 0.08 for similarity, while in the second case we have 0.89 for minability (less than the previous case) and 0.49 for similarity (more than the previous case). Thus, this confirms that there is no direct relation between these two indices, but both are helpful to address role elicitation by highlighting two different aspects of the user–permission set. Indeed, if the objective of role engineers is to elicit organizational roles, the similarity helps to identify the OUs where an organizational role that covers a relevant number of user–permission assignment exists. This happens, for instance, for 'Purchase Dept #4' because of the similarity of 0.69. On the other hand, if the objective of role engineers is to find functional roles, they have to consider the minability index. For example, the sub-branch Innovation is likely to be an easily solvable sub-problem due to a minability of 0.97, as confirmed by the low number of possible MERs.

As for risk indices, Table 2 highlights the behavior with respect to minability and similarity. For example, although the unit Audit has higher values for minability and similarity than the parent unit 'Quality', the high impact of the tasks performed by involved users compels a careful role design. According to this example, decomposing is a possible way to highlight data that requires particular attention from a risk management perspective. Another aspect to take into account is the required granularity for the partition. The most sensible approach is probably to stop decomposing when the risk indices do not increase or even increase slightly. For example, as shown in Table 1, by partitioning 'Product Development' according to its sub-units, the similarity-based risk index increases from 4.01 to 4.83, while the minability-based risk index grows from 0.92 to 1.05, reducing the gain in performing sub-unit driven analysis.

8. Concluding remarks

This paper describes a methodology that helps role engineers leverage business information during the role mining process. In particular, we demonstrate that by dividing data to analyze into smaller, more homogeneous subsets, it practically leads to more meaningful roles from a business perspective, hence decreasing the risk to make errors in managing them. To drive this process, two indices, referred to as minability and similarity, have been introduced. These indices are used to measure the expected complexity of analyzing the outcome of bottom-up approaches. In particular, we have shown how to apply such indices: to predict the effort needed to execute a role mining task over a set of user–permission assignments, thus being able to choose when to split a problem in several sub-problems; and, to select the top-down information that most simplifies the subsequent mining steps when more top-down information is available to role engineers. Leveraging these indices allows to identify the decomposition that increases business meaning in elicited roles in subsequent role mining steps, thus simplifying the analysis. We also introduced two fast probabilistic algorithms to efficiently compute such indices, making them also suitable for big organization with hundreds of thousands of users and permissions. The quality of the indices is also formally assured.

Several examples, developed on real data, illustrate how to apply the tools that implement the proposed methodology, as well as its practical implications. Achieved results support the quality and the viability of the proposal.

Acknowledgement

We would like to thank the anonymous reviewers that helped improve the quality of the paper.

References

[1] American National Standards Institute (ANSI) and InterNational Committee for Information Technology Standards (INCITS). ANSI/INCITS 359-2004, Information Technology—Role Based Access Control. 2004.
[2] Ebru Celikel, Murat Kantarcioglu, Bhavani Thuraisingham, Elisa Bertino, A risk management approach to RBAC, Risk and Decision Analysis 1 (2) (2009) 21–33 IOS Press.
[3] Alessandro Colantonio, Roberto Di Pietro, Alberto Ocello., A cost-driven approach to role engineering, Proceedings of the 23 rd ACM Symposium on Applied Computing, SAC '08, 2008, pp. 2129–2136.
[4] Alessandro Colantonio, Roberto Di Pietro, Alberto Ocello, Leveraging lattices to improve role mining, *Proceedings of the IFIP TC 11 23 rd International Information Security Conference, SEC '08* in IFIP International Federation for Information Processing, Springer-Verlag, 2008, pp. 333–347.
[5] Alessandro Colantonio, Roberto Di Pietro, Alberto Ocello, Nino Vincenzo Verde, A formal framework to elicit roles with business meaning in RBAC systems, Proceedings of the 14th ACM Symposium on Access Control Models and Technologies, SACMAT '09, 2009, pp. 85–94.
[6] Alessandro Colantonio, Roberto Di Pietro, Alberto Ocello, Nino Vincenzo Verde, A probabilistic bound on the basic role mining problem and its applications, *Proceedings of the IFIP TC 11 24th International Information Security Conference, SEC '09* in IFIP International Federation for Information Processing, Springer-Verlag, 2009, pp. 376–386.
[7] Alessandro Colantonio, Roberto Di Pietro, Alberto Ocello, Nino Vincenzo Verde., ABBA: adaptive bicluster-based approach to impute missing values in binary matrices, Proceedings of the 25th ACM Symposium on Applied Computing, SAC '10, 2010.
[8] Alessandro Colantonio, Roberto Di Pietro, Alberto Ocello, Nino Vincenzo Verde, Mining stable roles in RBAC. *Proceedings of the IFIP TC 11 24th International Information Security Conference, SEC '09*, IFIP International Federation for Information Processing, Springer-Verlag, 2009, pp. 259–269.
[9] Edward J. Coyne, Role-engineering, Proceedings of the 1st ACM Workshop on Role-Based Access Control, RBAC '95, 1995, pp. 15–16.
[10] Edward J. Coyne, John M. Davis., Role engineering for enterprise security management, Artech House (2007).
[11] Sabrina De Capitani Di Vimercati, Sara Foresti, Pierangela Samarati, Sushil Jajodia, Access control policies and languages, International Journal of Computational Science and Engineering 3 (2) (2007) 94–102 Inderscience Publishers.
[12] Reinhard Diestel, Graph Theory (Graduate Texts in Mathematics), 3 rd ed. Springer-Verlag, 2005.
[13] William Horne, Nikola Milosavljevic, Prasad Rao, Robert Schreiber, Robert E. Tarjan, Fast exact and heuristic methods for role minimization problems, Proceedings of the 13th ACM Symposium on Access Control Models and Technologies, SACMAT '08, 2008, pp. 1–10.
[14] Mario Frank and David Basin and Joachim M. Buhmann. A class of probabilistic models for role engineering.pages 299–310.
[15] M. P. Gallagher and A.C. O'Connor and B. Kropp. The economic impact of role-based access control. Technical report, Planning report 02-1, National Institute of Standards and Technology (NIST), 2002.
[16] Virgil Gligor. RBAC security policy model, preliminary draft report. Technical report, R23 Research and Development Department of the National Security Agency, 1995.
[17] Wassily Hoeffding, Probability inequalities for sums of bounded random variables, Journal of the American Statistical Association 58 (301) (1963) 13–30.
[18] Paul Jaccard, Etude comparative de la distribution florale dans une portion des Alpes et des Jura, Bulletin del la Société Vaudoise des Sciences Naturelles 37 (1901) 547–579.
[19] A. Kern, M. Kuhlmann, A. Schaad, J. Moffett, Observations on the role life-cycle in the context of enterprise security management, Proceedings of the 7th ACM Symposium on Access Control Models and Technologies, SACMAT '02, 2002, pp. 43–51.
[20] Martin Kuhlmann and Dalia Shohat and Gerhard Schimpf. Role mining – revealing business roles for security administration using data mining technology. pages 179–186.
[21] Haibing Lu, Jaideep Vaidya, Vijayalakshmi Atluri, Optimal Boolean matrix decomposition: application to role engineering, Proceedings of the 24th IEEE International Conferene on Data Engineering, ICDE '08, 2008, pp. 297–306.
[22] Ian Molloy, Ninghui Li, Tiancheng Li, Ziqing Mao, Qihua Wang, Jorge Lobo., Evaluating role mining algorithms, Proceedings of the 14th ACM Symposium on Access Control Models and Technologies, SACMAT '09, 2009, pp. 95–104.
[23] Gustaf Neumann, Mark Strembeck, A scenario-driven role engineering process for functional RBAC roles, Proceedings of the 7th ACM Symposium on Access Control Models and Technologies, SACMAT '02, 2002, pp. 33–42.
[24] M.E. Newman, S.H. Strogatz, D.J. Watts, Random graphs with arbitrary degree distributions and their applications, Phys Rev E Stat Nonlin Soft Matter Phys 64 (2) (2001).
[25] M.E. Newman, D.J. Watts, S.H. Strogatz, Random graph models of social networks, Proceedings of the National Academy of Sciences of the United States of America 99 (1) (2002) 2566–2572.
[26] H. Röckle, G. Schimpf, R. Weidinger., Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization, Proceedings of the 5th ACM Workshop on Role-Based Access Control, RBAC 2000, 2000, pp. 103–110.
[27] Thomas Schank, Dorothea Wagner, Approximating clustering coefficient and transitivity, Journal of Graph Algorithms and Applications (2005) 9.
[28] Jurgen Schlegelmilch and Ulrike Steffens. Role mining with ORCA. pages 168–176.
[29] Jaideep Vaidya, Vijayalakshmi Atluri, Qi. Guo, Nabil Adam, Migrating to optimal RBAC with minimal perturbation, Proceedings of the 13th ACM Symposium on Access Control Models and Technologies, SACMAT '08, 2008, pp. 11–20.
[30] Jaideep Vaidya and Vijayalakshmi Atluri and Janice Warner. RoleMiner: mining roles using subset enumeration. pages 144–153.
[31] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 440–442.
[32] Yang Xiang, Ruoming Jin, David Fuhry, Feodor F. Dragan., Succinct summarization of transactional databases: an overlapped hyperrectangle scheme, Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, 2008, pp. 758–766.
[33] Dana Zhang and Kotagiri Ramamohanarao and Tim Ebringer. Role engineering using graph optimisation. pages 139–144.

Alessandro Colantonio is currently an Identity Management & Security Specialist at Engiweb Security, a privately held company owned entirely by Engineering Ingegneria Informatica, an Italian-market leader in system and business integration and application management services. He is also a Ph.D. candidate in Mathematics at “Roma Tre” University, Roma, Italy. He received the Master's Degree in Computer Engineering with specialization in IT Systems and Applications at University of Pisa, Italy, in July 2001. He also received a Specialization Master in IT Security Management at “La Sapienza” University, Rome, Italy, in January 2008. He brings 8 years of experience in the security software industry. His main research interests include the identification of new and innovative methodologies, techniques, and models for risk management, role engineering, and data mining supporting the role lifecycle within Role-Based Identity & Access Management systems.

Roberto Di Pietro is currently an Assistant Professor at the Department of Mathematics of “Roma Tre” University, Roma, Italy. He is also with the UNESCO Chair in Data Privacy, Universitat Rovira i Virgili (Tarragona, Spain). He received the Ph.D. in Computer Science from the Università di Roma “La Sapienza”, Italy, in 2004. In 2004 he also received from the Department of Statistics of the same University a Specialization Diploma in Operating Research and Strategic Decisions. He received the Laurea degree in Computer Science from the University of Pisa, Italy, in 1994. To date, he has published more than 80 technical papers in high quality Conferences and Journals. His main research interests include: Role mining, security for mobile, ad-hoc, and underwater wireless networks, intrusion detection, security for distributed systems, secure multicast, applied cryptography and computer forensics.

Alberto Ocello is General Director at Engiweb Security. He holds a master of Electronic Engineering from the University of Rome, Italy and brings more than 25 years of experience in the security software industry. Alberto Ocello began his career at Page Europa (GTE Group) working in various roles in security application development, starting as Chief Product Architect and moving on to several product development groups in international military security projects. He then served as director of engineering, focusing on the application of PKI and new cryptographic technologies for enhancement of information security in electronic business scenarios. Alberto leads the company vision of role-based solutions, ensuring that technical excellence is the company cornerstone.

Nino Vincenzo Verde is currently a Ph.D. candidate in Mathematics at “Roma Tre” University, Roma, Italy. He received the Master's Degree in Computer Science at University “La Sapienza”, Rome, Italy, in September 2007. His main interests include access control and sensors/ad-hoc network security. In particular, he is involved in the definition of new models, tools and techniques that are useful for the definition of roles in Role-Based Access Control system. Also, he is currently working on the security of wireless and ad-hoc networks for the protection of Critical Infrastructures.