Interactive Role Mining Including Expert Knowledge into Evolutionary Algorithms

Simon Anderer[1], Nicolas Justen[1], Bernd Scheuermann[2] and Sanaz Mostaghim[3]
[1]*SIVIS GmbH, Grünhutstrasse 6, 76187 Karlsruhe, Germany*
[2]*Karlsruhe University of Applied Sciences, Karlsruhe, Germany*
[3]*Otto-von-Guericke Universität Magdeburg, Magdeburg, Germany*

Keywords: Dynamic Role Mining, User Interaction, Evolutionary Algorithms.

Abstract: To protect the security of corporate IT systems against erroneous or fraudulent behavior of employees, Role Based Access Control has proven to be an effective concept. The corresponding NP-complete Role Mining Problem aims at finding a minimal set of roles and an assignment of roles to users. A valuable source of additional information, which is not yet included in current role mining algorithms, is expert knowledge. Users of role mining software should be enabled to monitor the role mining process and interactively transfer their knowledge to the system, for example by suggesting good or deleting bad roles. This leads to dynamically occurring interaction events, which must be included into the optimization process preferably in real-time, since these have the potential to accelerate the optimization process or improve the solution quality. This paper introduces to interactive role mining. For this purpose, the hitherto static RMP is considered as dynamic optimization problem. Since evolutionary algorithms have shown to be a promising solution approach, it is shown how events emerging from user interaction can be integrated. The integration of different interaction events into the evolutionary algorithm and their impact on the optimization process are then evaluated in a range of experiments.

1 INTRODUCTION

IT systems of companies and organizations, in which several people collaborate, must be protected not only against external attacks such as malware and phishing, but also against erroneous or fraudulent employee behavior. According to a study by Pricewaterhouse-Coopers, more than half of all fraud cases, that could be identified within the participating companies, can be traced back either exclusively to internal perpetrators or to a combination of internal and external perpetrators (PwC, 2022). One approach to secure IT systems against such threats is Role Based Access Control (RBAC). At this, permissions, which correspond to the authorizations necessary to perform an operation on a data or business object, are not assigned to the users of an IT system directly, but are grouped into roles, which are then assigned to users (Ferraiolo and Kuhn, 1992). This process is also called *role engineering* and is carried out either in a top-down or bottom-up fashion. The top-down approach, in which company structures and business processes are analyzed in order to derive suitable roles, is very time-consuming and requires substantial human effort. Therefore, the bottom-up approach, in which role concepts are generated automatically, has gained more and more attention in recent years. The corresponding optimization problem is called the *Role Mining Problem* (RMP) and was shown to be NP-complete (Vaidya et al., 2007). It aims at finding a minimum number of roles based on a given assignment of permissions to users in order to enhance comprehensibility and manageability. One source of information, which can be of great value for the automatic creation of role concepts, but has not been considered in previous role mining literature, is expert knowledge. In practice, RBAC implementation projects are usually accompanied by experts to ensure that the implemented role concept meets established security standards and compliance rules. These experts have the ability to determine the suitability of certain roles included in role concepts depending on the company structure or the given assignments of permissions to users. Hence, experts should be able to monitor the role mining process, analyze the proposed role concepts and interactively suggest im-

Anderer, S., Justen, N., Scheuermann, B. and Mostaghim, S.
Interactive Role Mining Including Expert Knowledge into Evolutionary Algorithms.
DOI: 10.5220/0012153000003595
In *Proceedings of the 15th International Joint Conference on Computational Intelligence (IJCCI 2023)*, pages 151-162
ISBN: 978-989-758-674-3; ISSN: 2184-3236

provements. This results in dynamically occurring interaction events, which should be integrated into the optimization process, if possible, in real-time.

This paper aims at providing a framework to include expert knowledge into evolutionary algorithms (EAs). First, the Role Mining Problem is introduced in Section 2. In Section 3, the application of EAs to the RMP is discussed. Special attention is given to the *addRole-EA*, an evolutionary algorithm for the Basic RMP, which, due to its role-centric approach, has been selected as basis for the evaluation of different interaction events. A detailed overview of user interaction options in the context of role mining is provided in Section 4. Section 5 describes how dynamically occurring interaction events can be integrated into EAs in close to real-time. In Section 6, the inclusion of interaction events and their potential to enhance the optimization process is examined in different evaluation scenarios. Section 7 concludes the paper and presents paths for future research.

2 THE ROLE MINING PROBLEM

In the following, the main elements of the RMP are introduced:

- $U = \{u_1, u_2, ..., u_M\}$ a set of $M = |U|$ users.
- $P = \{p_1, p_2, ..., p_N\}$ a set of $N = |P|$ permissions.
- $R = \{r_1, r_2, ..., r_K\}$ a set of $K = |R|$ roles.
- $UPA \in \{0,1\}^{M \times N}$ the targeted permission-to-user assignment matrix, where $UPA_{i,j} = 1$ implies that permission p_j is assigned to user u_i.
- $UA \in \{0,1\}^{M \times K}$ a role-to-user assignment matrix.
- $PA \in \{0,1\}^{K \times N}$ a permission-to-role assignment matrix.
- $RUPA := UA \otimes PA \in \{0,1\}^{M \times N}$ the resulting permission-to-user assignment matrix, where $\otimes$ denotes the Boolean Matrix Multiplication: $(UA \otimes PA)_{i,j} = \max_{l \in \{1,...,k\}}(UA_{i,l} \cdot PA_{l,j})$.
- $\pi := \langle R, UA, PA \rangle$ a role concept.
- Π the set of all role concepts.

Based on this, the RMP can be defined as a matrix decomposition problem:

Basic Role Mining Problem. *Given a set of users U, a set of permissions P and a permission-to-user assignment matrix UPA, find a role concept π comprising a minimal set of Roles R, a corresponding role-to-user assignment matrix UA and a permission-to-role assignment matrix PA, such that each user is assigned exactly the permissions granted by UPA.*

$$\textbf{Basic RMP} = \begin{cases} \min & |R|, \\ \text{s.t.} & d(UPA, RUPA) = 0, \end{cases} \quad (1)$$

where $d(A,B) := \|A - B\|$ denotes the distance of two matrices $A, B \in \mathbb{R}^{m \times n}$ and $\|A\|$ denotes the sum of absolute values of elements of A: $\|A\| := \sum_{i=1}^{m} \sum_{j=1}^{n} |A_{i,j}|$.

A role concept π is called a feasible solution for the Basic RMP, if it satisfies the constraint in (1). In this case, π is also denoted *0-consistent*.

Since the RMP is a well-known optimization problem, a variety of established solution strategies has been proposed. Many approaches are based on the concept of permission grouping. Permissions are grouped into a set of candidate roles. These candidate roles are then assigned to users, mostly using greedy approaches (Blundo and Cimato, 2010; Molloy et al., 2009; Vaidya et al., 2010b). Another approach consists of mapping the RMP to other known problems in data science, e.g. the Minimum Tiling Problem (Vaidya et al., 2010a), the Minimum Biclique Cover Problem (Ene et al.,) or the Set Cover Problem (Huang et al., 2015), and using corresponding olution approaches. A detailed overview of different solution strategies is provided by Mitra (Mitra et al., 2016).

3 EVOLUTIONARY ALGORITHMS FOR THE RMP

In recent years, the application of evolutionary algorithms (EAs) in the context of the RMP has gained more and more attention. Since these are particularly well-suited for the integration of events emerging from user interaction, their application in the context of the RMP is considered in more detail at this point.

Each individual of an EA for role mining represents a possible solution of the RMP and thus a role concept. Therefore, its chromosome consists of a *UA* and a *PA* matrix. The set of roles *R* can be obtained from the rows of *PA*. An important differentiation criterion for EAs in the context of role mining is compliance with the 0-consistency constraint. Most role mining approaches based on EAs employ standard crossover and mutation methods such as one-point crossover and bitflip mutation. These lead to random changes in *UA* and *PA*, which correspond to random assignments of permissions to roles or of roles

to users, such that the 0-consistency constraint can be easily violated. Therefore, the corresponding EAs are not applicable for the Basic RMP (Saenko and Kotenko, 2011; Saenko and Kotenko, 2012; Saenko and Kotenko, 2016b; Du and Chang, 2014).

In (Anderer et al., 2020), the *addRole-EA* is presented, which comprises new, role-centric operators for crossover and mutation, that ensure compliance of the individuals of the addRole-EA with the 0-consistency constraint at all times within the optimization process. The role-centric approach of the addRole-EA is particularly well-suited for the integration of the interaction events considered in Section 6. A top level description of the addRole-EA including the values of its parameters is provided in Figure 1. Thereafter, the different components and methods of the addRole-EA are described.

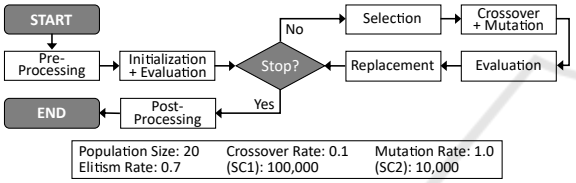

Figure 1: Integration of interaction events into addRole-EA.

Pre- and Post-Processing. In a first step of the addRole-EA, the size of the UPA matrix representing the RMP is reduced. For this purpose, four preprocessing steps have been identified comprising for example the removal of users that are assigned the same set of permissions except for one representative. The post-processing step adapts the obtained role concepts to the original problem size.

Initialization. In order to create an initial population, first a seed individual is created from $UA = UPA$ and $PA = I_N$, where I_N denotes the N-dimensional identity matrix. Since $UA \otimes PA = UPA \otimes I_N = UPA$, it complies with the 0-consistency constraint. Subsequently, according to the desired population size, new individuals are created from the seed individual by applying a random series of mutation operators.

Evaluation. Since the addRole-EA is an algorithm specifically tailored to the Basic RMP, the fitness of an individual equals the number of roles of the represented role concept.

Selection, Crossover and Mutation. At first, the individuals for crossover and mutation are selected randomly. Within crossover, roles of two individuals are exchanged to create offspring. Within mutation, new roles are created and added to the chromosome of an individual. In both cases the *addRole-method*, which constitutes the main method of the addRole-EA, is used to add a new role to the chromosome of an individual. At this, not only a new role is added, but the role structure is analyzed to identify and delete roles that have become obsolete through the addition of the new role. Figure 2 provides an example of the addRole-method. For better visibility, ones are displayed as black dots in UPA, UA and PA. Zeroes are represented as white dots.

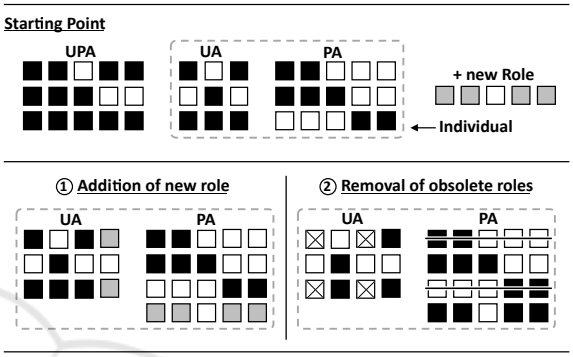

Figure 2: Example of addRole-method.

In this example, a new role, which is assigned permissions p_1, p_2, p_4 and p_5, is added to the individual. For this purpose, a new row is appended to PA representing the new role. In the first step, the new role is assigned to all users, that are assigned at least the same permissions. These are users u_1 and u_3. A new column is appended to UA representing the assignments of the new role to users. In the second step, it is checked, whether a role has become obsolete. For role r_1, which assigns p_1 and p_2 to u_1 and u_3, these assignments are now also covered by the new role. The same is valid for the assignments of permissions to users induced by r_3. Hence, these roles are deleted from the individual such that in total the number of roles could be reduced by one.

Replacement. The addRole-EA is a steady-state evolutionary algorithm. For replacement an elitist selection scheme is applied. First, based on the elitism rate, individuals for the next generation are selected based on their fitness. The remaining individuals are then selected randomly to ensure diversity among the individuals of the next generation's population.

Stopping Condition. The addRole-EA is terminated according to two criteria: either a maximum number of iterations has been executed (SC1), or no further improvement has been achieved within a given number of iterations (SC2).

4 USER INTERACTION IN ROLE MINING

Although it seems natural to integrate expert knowledge into ongoing bottom-up role mining processes, to enhance the quality of the resulting role concepts, or to accelerate the optimization process, to the best of our knowledge no research has been conducted in this area so far. Some approaches consider the integration of dynamically occurring events into the role mining process. However, these events stem from changes in the company structure, such as new employees joining a company and employees leaving a company (Anderer et al., 2021a), or from changes in the assignment of permissions to users (Saenko and Kotenko, 2016a). Events caused by user interaction are not considered.

König and Schneider examine possibilities of how users of optimization software can interact with an optimization algorithm used for automatic generation of layouts. In order to avoid confusion with regular users of IT systems in the context of role mining, such users of optimization software will be referred to as decision makers (DM) throughout the remainder of this paper. Considering automatic layout generation, DMs are provided with the option to move, scale, add and remove elements during the layout generation process. Furthermore, they distinguish between direct and indirect manipulations. Direct manipulations correspond to a modification of solution candidates. Indirect manipulations include the modification of optimization objectives or constraints as well as the adaptation of parameters of the applied optimization algorithm (König and Schneider, 2011).

Nascimiento examined the interaction of a DM with evolutionary algorithms for different graph-based problems. For the Graph Clustering Problem, the DM is given the option to either destroy clusters or merge two clusters. For the Graph Drawing Problem, the DM is provided with the possibility to move vertices. For the Map Labeling Problem, the DM is given the option to exchange two vertices. Furthermore, the DM is allowed for dynamic focusing of the optimization on manually chosen sub-problems. Finally, interaction possibilities aiming at the specifications of evolutionary algorithms, like deliberate inclusion of certain individuals into the population of an evolutionary algorithm are presented (Do Nascimento, 2003).

Most of the current research focuses on the area of indirect interactions. The goal is to help the EA find the regions of interest on the pareto front for the DM. For this purpose many different methods have been developed in the past. An overview is provided by Xin et al. in (Xin et al., 2018)

Table 1: Interaction events in the context of role mining.

ID	Event
I01	*Add a new role*
I02	*Delete an existing role*
I03	*Merge roles* (Create a new role from two existing roles)
I04	*Split roles* (Create two new roles from one existing role)
I05	*Edit PA* (whilst preserving 0-consistency)
I06	*Edit UA* (whilst preserving 0-consistency)
I07	*Adapt mutation rate*
I08	*Adapt crossover rate*
I09	*Adapt population size*
I10	*Adapt replacement parameters*
I11	*Adapt stopping condition*
I12	*Focus on selected users and corresponding role assignments* (e.g. using specific mutation or crossover operators)
I13	*Exclude selected users* (Exclude users (and corresponding role assignments) from the optimization process)
I14	*Store a role concept* (Transfer an individual from the current population into the role concept repository)
I15	*Insert a role concept into the population* (Insert an individual of the role concept repository into the current population)

In the following, an overview of user interaction events relevant in the context of role mining is provided. These events are given identifiers I01 to I15 and are grouped into four categories, see Table 1. The first category (I01-I06) contains interaction events leading to a direct manipulation of role concepts. Roles and the corresponding assignment to users can be edited, added or removed. A DM can try to include his or her expert knowledge into the optimization process, for example by adding roles that have proven to be particularly good in the past (I01) or by removing roles that seem unpromising (I02). However, it is not guaranteed that these suggestions are also well-suited in the current role mining scenario. It should therefore be possible for the optimization algorithm used to reverse the proposed changes later on in the optimization process, if the expected improvement is not obtained.

The second category (I07-I11) contains adjustments of the parameters of the role mining algorithm used. Since these indirect interactions strongly depend on the role mining approach used, some examples for EAs are provide at this point.

Analogous to the approach of Nascimiento, the DM should be provided with the option to set the focus of the optimization algorithm to certain areas. Such events are contained in the third category (I12-I13). For example, the role mining algorithm could be prompted to focus particularly on the users of a certain department of the company if no satisfactory roles for the users of this department are included in the proposed role concepts yet (I12). Addition-

ally, in order to reduce the problem size, a DM can decide to exclude users and the corresponding roles from further optimization (I13). Since this means that the roles assigned to the excluded users can no longer be modified, which in turn implicitly affects the further optimization process, this interaction possibility should be treated with great foresight and caution.

The fourth category (I14-I15) addresses the prevailing risk of EAs to get stuck in local optima. Individuals obtained in previous iterations can be stored and reintegrated into the population of the evolutionary algorithm, whenever necessary, thereby avoiding the necessity of a complete restart of the optimization process. In particular, in dynamic optimization, the fitness landscape is subject to change over time. Hence, it is possible that some of the stored individuals may have better fitness values than the individuals of the current population. Therefore, injecting such stored individuals into the current population appears to be a promising approach. This is also referred to as memory-based evolutionary algorithms and has been covered in previous publications. A survey on memory-based evolutionary algorithms is provided by Branke (Branke, 1999). Branke proposes to adopt the concept of memory-based evolutionary algorithms to the domain of role mining with user interaction. In this scenario, the DM can store interesting role concepts within a so-called role concept repository (I14). Hence, the DM is able to analyze and possibly deploy these role concepts later (I15).

5 INTEGRATION OF INTERACTION EVENTS INTO EVOLUTIONARY ALGORITHMS

One advantage of EAs considering the inclusion of dynamically occurring interaction events is their iterative procedure. At the beginning of each iteration, it can be checked whether one or more events are currently pending. If this is the case, the corresponding event-handling methods can be executed. Figure 3 shows the alteration of the sequential process of the addRole-EA for the integration of event-handling methods.

In addition, it is recommended not to terminate the execution of the addRole-EA according to (SC1) or (SC2), but to continue the role mining process, whenever computing capacity is available. In this way, the role concepts can potentially be further improved and events can be processed close to real-time. Since this approach is independent of the specific features of the addRole-EA, it can be transferred to any other evolutionary algorithm.

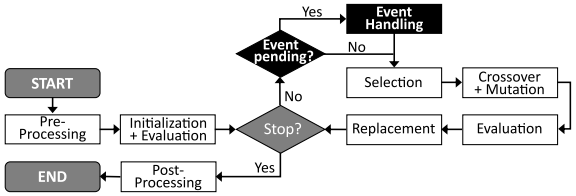

Figure 3: Integration of interaction events into addRole-EA.

6 EVALUATION AND EXPERIMENTS

In this section, the events of the first event category defined in Section 4 are examined in more detail, as these include direct manipulations of individuals and therefore impose a particular challenge from an algorithmic point of view. Special focus is given to interaction event I01, where a DM proposes potentially good roles, as well as I02, where a DM removes potentially bad roles, as changes resulting from the other events of this category can be implemented as a combination of adding and removing roles. For this purpose, it is first shown how established benchmarks for the RMP can be extended to be suitable for the simulation of event I01 and I02 and their evaluation. Subsequently, the two events selected are examined in different evaluation scenarios.

6.1 Simulation of Events and Preparation of Benchmarks

In practice, the deployment of role concepts is often accompanied by experts and consultants. Over time, these experts acquire extensive knowledge in the area of role mining and are therefore able to identify roles that bear the potential to improve the role concepts encoded by the individuals of the current population or to accelerate the optimization process in certain situations. Such roles will be referred to as *good* roles in the following. Likewise, they are able to assess which roles potentially hamper the optimization process. These roles will be referred to as *bad* roles.

It is evident that it is not possible to transfer this knowledge onto the synthetic evaluation scenarios corresponding to available benchmark instances for the RMP. Therefore, other methods must be found in order to identify *good* and *bad* roles in a given benchmark instance. For the evaluation of the interaction events in this paper, three benchmark instances of *RMPlib*, a publicly accessible library for role mining

Table 2: Benchmark instances selected for evaluation.

	Users M	Prms. N	Roles K_0	Density ρ
PS_02	50	50	25	0.240
PS_05	100	100	50	0.137
PM_01	500	500	150	0.062

benchmarks, were selected. The benchmark instance *PLAIN_small_02 (PS_02)* is a rather small data set, which has a comparatively high density. The benchmark instance *PLAIN_small_05 (PS_05)* is slightly larger, but less densely populated. The benchmark instance *PLAIN_medium_01 (PM_01)* is a medium-size, low-density data set. The specifications of the instances selected are displayed in Table 2. At this, ρ denotes the density of the *UPA* matrix corresponding to the respective benchmark instance. Furthermore, K_0 denotes the number of roles that were used to create the benchmark and thus serves as upper bound on the minimum number of roles. For a more detailed description of the benchmark instances of *RMPlib*, refer to (Anderer et al., 2021b).

In order to identify good roles, the addRole-EA was run 200 times on each benchmark instance. Subsequently, the 20 best role concepts obtained for each instance were selected. Based on these results, a role was classified as a *good* role, if it was included in each of the 20 role concepts. From this, a set of 10 *good* roles, each of which are assigned between 3 and 8 permissions, was obtained for PS_02. For PS_05, the set of *good* roles is comprised of 46 good roles, each of which are assigned between 2 and 11 permissions. For PM_01, 144 *good* roles could be identified, ranging from 3 to 22 permissions each. A role was classified a *bad* role, if it was not included in at least one of the 20 best role concepts. Even though the presented procedure was solely applied to the three benchmark instances selected, it can be applied to the other benchmark instances of *RMPlib* as well. At this point, it must be noted that, in the context of *RMPlib*, the addRole-EA is already used to identify *good* and *bad* roles before the execution of the role mining process. This does not correspond to practice, where this knowledge emerges from the growing wealth of experience of role mining experts.

6.2 Event I01: Addition of *Good* Roles

In the following, it is examined whether the addition of the *good* roles defined in the last section can actually enhance the optimization process. For this purpose, random roles are selected from the set of *good* roles at different points in time during the optimization process. Here, each *good* role selected induces an instance of interaction event I01, which is handled

Table 3: Parameter values for the evaluation of I01.

	PS_02	PS_05	PM_01
Number of events $\|E\|$	3; 5	5; 10	20; 30
Simulated at iteration t_i	$t_1 = 5{,}000$;	$t_2 = 10{,}000$;	$t_3 = 50{,}000$

by adding the selected role to all individuals in the current population using the addRole-Method of the addRole-EA. Should it occur that multiple instances of event I01 are pending at the same time, the corresponding *good* roles are added to the individuals successively by repeatedly calling the addRole-method. To establish comparability, a copy of the population is created immediately before the *good* roles are added. This copy, which is referred to as Pop_0, is further optimized without including the instances of interaction event I01. The population in which the instances of I01 are included is referred to as Pop_{I01}.

The number of *good* roles, which are transferred into the optimization process, is based on the benchmark instance used. In each case, up to 20% of the number of roles used to create the benchmark instance were selected randomly from the set of *good* roles and added to the individuals of the current population at $t_1 = 5{,}000$, $t_2 = 10{,}000$ and $t_3 = 50{,}000$. Table 3 shows the number of instances $|E|$ of interaction event I01 as well as the timing of the event simulation resulting in the different evaluation scenarios. The tests for each evaluation scenario were repeated 20 times with different random seeds and run on a computer with the following specifications: processor Intel Core i5-4570S, 2.90 GHz, 16 GB RAM. The parameters of the addRole-EA were adopted from (Anderer et al., 2020), see Figure1.

Figures 4 - 6 show the progression of the number of roles over iterations $r_0^*(t)$ and $r_{I01}^*(t)$, which refer to the number of roles of the best individual of Pop_0 resp. Pop_{I01} at iteration t, where five *good* roles were added at t_1 on PS_02, PS_05 and PM_01. Since similar effects are observed for all other combinations of parameter values on the selected benchmark instances, only one representative graph is shown for each instance in the following.

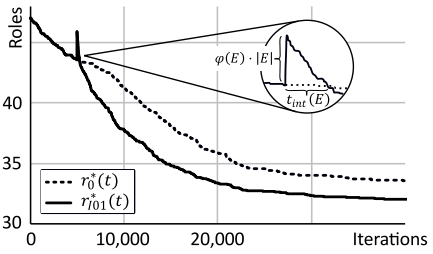

Figure 4: Number of roles over iterations for I01 on PS_02.

Table 4: Evaluation of the addition of $|E|$ *good* roles.

		$t_1 = 5{,}000$				$t_2 = 10{,}000$				$t_3 = 50{,}000$			
		$r_0^*(\hat{t})$	$r_{I01}^*(\hat{t})$	$\varphi(E)$	$t_{int}(E)$	$r_0^*(\hat{t})$	$r_{I01}^*(\hat{t})$	$\varphi(E)$	$t_{int}(E)$	$r_0^*(\hat{t})$	$r_{I01}^*(\hat{t})$	$\varphi(E)$	$t_{int}(E)$
PS_02	$\|E\| = 3$	31.30	30.95	0.58	330	30.85	31.65	0.27	250	30.35	30.40	0.05	10
	$\|E\| = 5$	33.55	32.05	0.47	280	30.85	30.40	0.30	150	30.10	30.15	0.10	40
PS_05	$\|E\| = 5$	50.15	50.45	0.22	90	50.30	50.10	0.04	50	50.55	50.55	-0.05	0
	$\|E\| = 10$	50.80	50.15	0.21	100	49.95	50.30	0.03	20	49.95	49.95	0.00	0
PM_01	$\|E\| = 20$	153.25	153.00	-0.02	0	153.00	152.70	-0.33	0	151.80	151.65	-0.05	0
	$\|E\| = 30$	152.70	152.55	-0.10	0	152.45	152.25	-0.38	0	151.85	151.65	-0.00	0

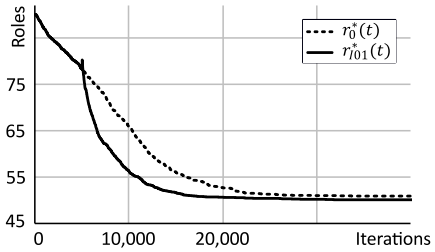

Figure 5: Number of roles over iterations for I01 on PS_05.

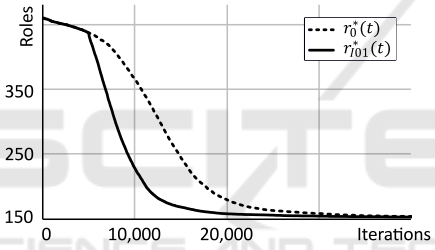

Figure 6: Number of roles over iterations for I01 on PM_01.

First, it can be seen that the integration of event instances significantly improves the optimization process on all benchmark instances. On PS_02 and PS_05, however, the integration of interaction events results in a temporary increase in the number of roles of the individuals in population Pop_{I01}. Since the occurrence of interaction events is disregarded in the control population Pop_0, the number of roles of its individuals remains unchanged. To provide a measure for this effect, a new key figure, the impact $\varphi(E)$, is introduced for a set of interaction events E. It is defined as the difference in the number of roles of the best individual at the time immediately after application of the corresponding event-handling method t^+ and directly before event occurrence t divided by the total number of event instances:

$$\varphi(E) := \frac{r(I^*, t^+) - r(I^*, t)}{|E|} \tag{2}$$

A further interesting key figure to describe the course of the optimization process comparing Pop_0 and Pop_{I01} is the number of iterations $t_{int}(E)$ needed until the fitness of the best individual in Pop_{I01} equals the fitness of the best individual of Pop_0 for the first time after the occurrence of the interaction event. In case $\varphi(E) > 0$, this corresponds to the first intersection of both curves after event occurrence, see Figure 4, and is obtained as follows:

$$t_{int} = min\{t - t_i : \; r(I_{I01}^*, t) = r(I^*, t)\} \tag{3}$$

where t_i corresponds to the iteration, in which the *good* roles were added to the chromosomes of the individuals of Pop_{I01}. In case $\varphi(E) \leq 0, t_{int} := 0$. Table 4 shows the values obtained for $t_{int}(E)$, the impact $\varphi(E)$ as well as the number of roles $r_0^*(\hat{t})$ and $r_{I01}^*(\hat{t})$ of the best individuals of both populations after executing $\hat{t} = 100{,}000$ iterations of the addRole-EA. For easier comprehension, the impact $\varphi(E)$ as well as $t_{int}(E)$ are illustrated in Figure 4.

According to the observations in Figures 4 and 5, it can be seen that on PS_02 and PS_05 the values of the impact are mostly positive. This results from the fact that new roles are added to the individuals by instances of event I01. Even though it seems logical that an instance of event I01, which corresponds to the addition of one role, would increase the number of roles by one, the values for the impact range between -0.38 and 0.58 and are thus significantly smaller than one or even negative. This is caused by the operation principle of the addRole-method. As shown in Section 3, adding a new role by means of the addRole-method includes the deletion of existing roles that have become obsolete. Since in this test case, *good* roles were added, it can be assumed that existing roles became obsolete in a relatively large number of cases. On PM_01, this even leads to negative values of $\varphi(E)$, which corresponds to an immediate improvement of the number of roles due to the integration of interaction events. Additionally, it can be seen that the simulation of interaction events at later points in time causes a reduced impact on PS_02 and PS_05. This may be due to the definition of good roles used in this paper. Whenever *good* roles are added rather at the end of the optimization process, it is possible that they are already included in some of the individuals, so that these individuals are not affected by the addition

Table 5: Iterations and time (s) for I01 on PS_02.

	Number of iterations needed			*Computation time (s) needed*		
k	$\|E\|=0$	3	5	$\|E\|=0$	3	5
40	7,370	3,190	2,700	154.62	63.91	53.14
38	10,375	5,270	4,390	204.94	99.68	82.37
36	13,645	8,140	7,920	252.76	143.69	136.23
34	20,700	10,850	12,580	344.45	180.15	198.24
32	-	20,340	-	-	291.11	-

Table 6: Iterations and time (s) for I01 on PS_05.

k	$\|E\|=0$	5	10	$\|E\|=0$	5	10
75	1,350	430	260	15.78	3.21	4.35
70	3,180	1,470	790	35.48	13.88	9.88
65	4,995	2,690	1,580	53.41	25.33	17.55
60	6,630	4,450	3,300	67.82	40.44	33.18
55	9,860	7,150	5,960	93.98	61.65	54.71

of the roles. In all test setups, that result in positive impact values, the rather small values of $t_{int}(E)$ show that, although the number of roles $r^*_{I01}(t)$ initially increases due to event integration, it is again at least as good as $r^*_0(t)$ within a few iterations. Thus, the added roles require a certain amount of time to reveal their positive effects on the optimization process.

Table 4 further illustrates that the inclusion of Instances of I01 in Pop_{I01} leads to better results with respect to the number of roles in 11 out of 18 cases at $\hat{t} = 100,000$ compared to only 5 out of 18 cases, in which better results were obtained for Pop_0.
However, the potential of interaction event I01 lies mainly in the short-term effects. In Figures 4 - 6, it could already be seen that the inclusion of instances of event I01 leads to a significantly faster reduction of the number of roles of the best individual of the associated population Pop_{I01} compared to that of the control population Pop_0. In order to examine this in more detail, for each benchmark instance, different role levels k were specified, which serve as reference values to evaluate the short-term performance of the inclusion of I01. Tables 5 - 7 show the number of iterations and the computation time needed to attain the respective role level from the occurrence of the interaction events at t_1. Again, similar effects are obtained considering t_2 and t_3, so that the corresponding tables are omitted at this point.

Table 7: Iterations and time (s) needed for I01 on PM_01.

	Number of iterations needed			*Computation time (s) needed*		
k	$\|E\|=0$	20	30	$\|E\|=0$	20	30
420	1,750	350	300	327,32	52,41	54,80
400	3,045	1,020	770	548,52	156,03	126,99
380	4,155	1,640	1,180	722,81	244,01	184,51
360	5,145	2,160	1,590	863,79	313,47	237,29
340	5,970	2,680	2,020	973,36	378,06	289,43

It turns out that in all cases, the addition of good roles results in the specified role levels being achieved in significantly fewer iterations as well as in significantly less computation time. Furthermore, it seems that the more good roles are added, the fewer iterations and computation time is needed. In conclusion, it can therefore be stated that the inclusion of expert knowledge, by means of interaction event I01, has the potential to significantly accelerate the optimization process in short-term consideration and leads to better results in long-term consideration in many cases.

6.3 Event I02: Deletion of *Bad* Roles

In this section, analogous to the study of adding *good* roles, it is examined how a DM can transfer his or her expert knowledge into the role mining process by removing potentially bad roles. However, while the addRole-EA already provides a method for the addition of *good* roles using the addRole-method, the removal of *bad* roles is not readily supported due to the 0-consistency constraint. If *bad* roles were simply deleted without further action, users could lose permissions essential to performing their tasks. Therefore, suitable repair methods must be developed to ensure compliance with the 0-consistency constraint after roles were removed. For this purpose, users lacking permissions according to UPA are identified in a first step. Subsequently, one of the following repair methods is executed:

R1: Assign all Roles + Unique Roles. In order to reassign permissions to the users, which were withdrawn by the removal of *bad* roles, first, it is investigated whether some of the existing roles in PA can be assigned to the users without violation of the 0-consistency constraint to cover some of the required permissions. If, thereafter, a user is still lacking permissions, a new role is created for each of these permissions. This corresponds to the idea of the initialization method of the addRole-EA. An advantage of this method is that the algorithm gains the possibility to create new roles from scratch. However, many additional roles are required to compensate for the deletion of *bad* roles using this approach.

R2: Assign all Roles + One Role + Unique Roles. In order to avoid the creation of many new roles, repair method R2 aims at grouping the uncovered permissions into one role. Again, the users are assigned all roles available in PA, that can be assigned to them without violation of the 0-consistency constraint. If, thereafter, a user is still lacking permissions, a new

role is created from the remaining uncovered permissions. If this role is not equal to the *bad* role, which was previously removed, it is assigned to the users. If this role equals the *bad* role, analogous to R1, a new role is created for each of the uncovered permissions. An advantage of this method is that the creation of many new roles can possibly be avoided. An algorithmic description on how the UA and PA matrices of an individual I are adapted to the removal of a bad role r_{bad} by repair method R2 is provided in Algorithm 1. If lines 9 - 17 are replaced by lines 13 - 16, one obtains repair method R1.

Algorithm 1: *Repair Method R2.*

```
    Input:  UPA, UA, PA, r_bad
    Output: UA, PA
 1  for each role r corresponding to a row of PA do
 2      for each user u ∈ U do
 3          if r can be assigned to u without
              violation of 0-consistency then
 4              Assign r to u;
 5          end
 6      end
 7  end
 8  Identify uncovered permissions by comparing
      UPA and RUPA;
 9  Create new role r_new, which is assigned all
      uncovered permissions;
10  if new role does nor equal bad role r_new ≠ r_bad
      then
11      Add r_new to individual I using
          addRole-method;
12  else
13      for each uncovered permission p do
14          Create new role r, which is assigned p;
15          Add r to individual I using
              addRole-method;
16      end
17  end
```

Evaluation. The evaluation of interaction event I02 was conducted analogously to the evaluation of event I01. The parameters on which the various evaluation scenarios are based can therefore also be found in Table 3. At the moment of event occurrence, three identical copies of the current populations are made. In Pop_{R1} respectively Pop_{R2}, for each individual, each role is examined individually and removed if it was classified a *bad* role as described in Section 6.1. This is repeated until either all roles of the individual are examined or the maximum number of roles to be deleted from an individual, as specified by $|E|$, is attained. Subsequently, the compliance of the individuals with the 0-consistency constraint is restored once using repair method R1 and once using R2. Again, Pop_0 serves as control population in which I02 is disregarded.

Figure 7 shows the progression of the number of roles over iterations $r_0^*(t)$, $r_{R1}^*(t)$ and $r_{R2}^*(t)$ which denote the number of roles of the best individual of Pop_0, Pop_{R1} and Pop_{R2} at iteration t, where up to five *bad* roles were removed at t_1 on PS_02.

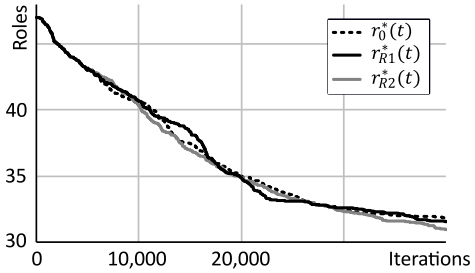

Figure 7: Number of roles over iterations for I02 on PS_02.

It can be seen that event I02, unlike I01, does not lead to any significant improvements in the optimization process either by means of repair method R1 or R2. The same can be observed considering t_1 and t_2 on all of the selected benchmark instances and can be explained by the operation principle of the addRole-EA. By repeatedly adding new roles and subsequently deleting obsolete roles using the addRole-method, it can be assumed that bad roles will be eliminated automatically. This is also evident when considering the long-term effects of I02. For this purpose, Table 8 shows the number of roles of the best individual of the three populations at $\hat{t} = 100,000$ as well as the impact of the interaction events $\varphi_{R1}(E)$, when using repair method R1 respectively $\varphi_{R2}(E)$ when using R2. At t_1 and t_2, in only 4 out of 12 cases, the results obtained from event integration obtained better or the same results for both repair methods compared to the results obtained for Pop_0. In all the other cases, at least one of the repair methods provided worse results compared to Pop_0. This suggests that the removal of *bad* roles at the beginning of the optimization process has no significant effect on the optimization process which is also underlined by the corresponding low impact values. At t_3, however, the integration of interaction events leads to better results in all cases independent of whether R1 or R2 is used. This may be due to the fact that at the time of event occurrence, the addRole-EA has almost converged on all benchmark instances. The removal of *bad* roles and the subsequent addition of new, smaller roles causes for a modification of the individuals and possibly creates new optimization potential. Table 8 shows that R2 leads to better results compared to R1 in 5 out of 6 cases, which is possibly due to the fact that R2 first attempts to create one single role to cover the uncov-

Table 8: Evaluation of the deletion of $|E|$ *bad* roles.

		$t_1 = 5{,}000$					$t_2 = 10{,}000$					$t_3 = 50{,}000$				
		$r_0^*(\hat{t})$	$r_{R1}^*(\hat{t})$	$r_{R2}^*(\hat{t})$	$\varphi_{R1}(E)$	$\varphi_{R2}(E)$	$r_0^*(\hat{t})$	$r_{R1}^*(\hat{t})$	$r_{R2}^*(\hat{t})$	$\varphi_{R1}(E)$	$\varphi_{R2}(E)$	$r_0^*(\hat{t})$	$r_{R1}^*(\hat{t})$	$r_{R2}^*(\hat{t})$	$\varphi_{R1}(E)$	$\varphi_{R2}(E)$
PS_02	$\|E\| = 3$	30.35	29.60	30.40	-0.03	-0.02	29.50	29.90	29.90	0.07	0.03	30.00	29.80	29.70	1.12	0.43
	$\|E\| = 5$	30.65	30.60	29.55	0.03	0.04	30.60	30.30	30.60	0.16	0.13	29.15	29.25	29.05	1.23	0.63
PS_05	$\|E\| = 5$	50.25	49.90	50.15	0.00	0.00	49.80	49.85	50.35	0.24	0.17	50.45	50.00	49.70	0.93	0.23
	$\|E\| = 10$	50.15	50.45	50.10	0.03	0.02	50.05	50.05	50.30	0.46	0.36	50.45	49.80	49.70	0.54	0.23
PM_01	$\|E\| = 20$	151.85	151.55	151.25	0.00	0.00	151.35	151.10	152.10	0.07	0.03	151.65	150.55	150.50	0.41	0.41
	$\|E\| = 30$	151.55	151.95	151.90	0.00	0.00	151.30	151.60	151.15	0.13	0.05	151.70	151.30	151.80	0.68	0.64

ered permission after the removal of *bad* roles, such that less new roles are created in total.

A further interesting observation is that the impact values tend to increase, the later the instances of I02 occur. This is because roles are assigned rather few permissions at the beginning of the optimization process due to the initialization method of the addRole-EA. These can therefore be used more frequently to cover needed permissions which were withdrawn from users removing *bad* roles. Toward the end of the optimization process, roles are more likely to be assigned a larger number of permissions. Consequently, a larger number of new roles is necessary to cover the needed permissions.

In summary, it can be concluded that, especially in comparison with interaction event I01, the deletion of *bad* roles does not have a major impact on the optimization process. However, at later points within the optimization process, the removal of *bad* roles has proven its potential to further improve the proposed role concepts.

7 CONCLUSION AND FUTURE WORKS

This paper investigated the integration of expert knowledge into evolutionary algorithms for role mining. First, an overview of events relevant in the context of role mining, which mostly occur dynamically during the optimization process and are triggered by the interaction of a decision maker with role mining software, was provided. A framework was presented on how to integrate interaction events in near real-time into the iterative optimization process of evolutionary algorithms. Additionally, a method designed to enable the simulation of interaction events using known role mining benchmarks was presented. Based on this method, different events and their effects on the optimization process were evaluated in a series of experiments. It was demonstrated that the integration of expert knowledge, especially by adding potentially good roles, leads to better optimization results and thus better role concepts in many cases. Furthermore, it causes for a significant acceleration of the optimization process. This is of particular importance in large companies, where several thousand users work together in an ERP system and therefore long runtimes of the role mining algorithm are to be expected. Even if the interaction event corresponding to the deletion of potentially bad roles has a significantly lower impact on the optimization process compared to the addition of potentially good roles, it could be shown that this can still improve the optimization result, especially if these are deleted rather at a later point in time within the optimization process.

The definition of potentially good roles in this paper ensured that these provide a certain quality in the context of the respective benchmark instance. In practice, however, it is possible that assumingly good roles proposed by a decision maker have no or even negative influence on the optimization process in the scenario under consideration. It is therefore desirable to maintain both, individuals that have been modified due to the interactions of a decision maker as well as unmodified individuals, in the population for a certain time. However, it has been shown that in many cases adding *good* roles initially increases the number of roles of an individual (positive impact). Due to the elitist replacement method of the addRole-EA, these individuals would be inclined not to be transferred to the following generations. Analogously, in case of negative impact values, the original, unmodified individuals would potentially not be transferred to the subsequent generations. One way to address this is to develop suitable survival strategies that ensure that both modified and unmodified individuals survive long enough to unfold their potential in terms of improving the optimization process. Furthermore, the cooperation between users and the evolutionary role mining algorithm could adaptively be improved by the introduction of a role repository where experts can store potentially good roles, so that the addRole-EA can propose them autonomously in future role

mining projects, e.g. via customized mutation and crossover methods. Since the presented methods for the inclusion of expert knowledge have specifically been designed for use in practice, an application in industrial use case scenarios would be desirable.

ACKNOWLEDGEMENTS

This study was funded by the German Ministry of Education and Research under grant number 16KIS1000.

REFERENCES

Anderer, S., Kempter, T., Scheuermann, B., and Mostaghim, S. (2021a). The dynamic role mining problem: Role mining in dynamically changing business environments. In Bäck, T., Wagner, C., Garibaldi, J. M., Lam, H. K., Cottrell, M., Merelo, J. J., and Warwick, K., editors, *Proceedings of IJCCI 2021, Online Streaming, October 25-27, 2021*, pages 37–48. SCITEPRESS.

Anderer, S., Kreppein, D., Scheuermann, B., and Mostaghim, S. (2020). The addRole-EA: A new evolutionary algorithm for the role mining problem. In Guervós, J. J. M., Garibaldi, J. M., Wagner, C., Bäck, T., Madani, K., and Warwick, K., editors, *Proceedings of IJCCI 2020, Budapest, Hungary, November 2-4, 2020*, pages 155–166. SCITEPRESS.

Anderer, S., Scheuermann, B., Mostaghim, S., Bauerle, P., and Beil, M. (2021b). RMPlib: A library of benchmarks for the role mining problem. In Lobo, J., Pietro, R. D., Chowdhury, O., and Hu, H., editors, *Proceedings of SACMAT 2021, Virtual Event, Spain, June 16-18, 2021*, pages 3–13. ACM.

Blundo, C. and Cimato, S. (2010). A simple role mining algorithm. In Shin, S. Y., Ossowski, S., Schumacher, M., Palakal, M. J., and Hung, C.-C., editors, *Proceedings of the 2010 ACM SAC, Sierre, Switzerland, March 22-26, 2010*, pages 1958–1962, New York, New York. ACM.

Branke, J. (1999). Memory enhanced evolutionary algorithms for changing optimization problems. In *Proceedings of IEEE CEC 1999, Washington, DC, USA July 6-9, 1999*, volume 3, pages 1875–1882. IEEE.

Do Nascimento, H. A. D. (2003). *User Hints for Optimisation Processes*. PhD thesis, University of Sydney. Information Technologies.

Du, X. and Chang, X. (2014). Performance of AI algorithms for mining meaningful roles. In *Proceedings of IEEE CEC 2014, Beijing, China, July 6-11, 2014*, pages 2070–2076. IEEE.

Ene, A., Horne, W. G., Milosavljevic, N., Rao, P., Schreiber, R., and Tarjan, R. E. Fast exact and heuristic methods for role minimization problems. In Ray, I. and Li, N., editors, *Proceedings of SACMAT 2008, Estes Park, CO, USA, June 11-13, 2008*, pages 1–10, New York, New York. ACM.

Ferraiolo, D. F. and Kuhn, D. R. (1992). Role-based access controls. In *Proceedings of NCSC 1992, Baltimore, Maryland, USA, October 13-16, 1992*, pages 554 – 563, Baltimore.

Huang, H., Shang, F., Liu, J., and Du, H. (2015). Handling least privilege problem and role mining in RBAC. In *Journal of Combinatorial Optimization*, volume 30, pages 63–86.

König, R. and Schneider, S. (2011). Nutzerinteraktion bei der computergestützten generierung von layouts. In Donath, D. and König, R., editors, *Arbeitspapiere Nr. 8 Informatik in der Architektur*, Weimar. Bauhaus-Universität Weimar.

Mitra, B., Sural, S., Vaidya, J., and Atluri, V. (2016). A survey of role mining. In *ACM Computing Surveys*, volume 48, pages 1–37, New York, New York. ACM.

Molloy, I. M., Li, N., Li, T., Mao, Z., Wang, Q., and Lobo, J. (2009). Evaluating role mining algorithms. In Carminati, B. and Joshi, J. B. D., editors, *Proceedings of ACM SACMAT 2009, Stresa, Italy, June 3-5, 2009*, pages 95–104, New York, New York. ACM.

PwC (2022). *PwC's Global Economic Crime and Fraud Survey 2022*. PricewaterhouseCoopers.

Saenko, I. and Kotenko, I. (2011). Genetic algorithms for role mining problem. In Cotronis, Y., Danelutto, M., and Papadopoulos, G. A., editors, *Proceedings of PDP 2011, Ayia Napa, Cyprus, 9-11 February 2011*, pages 646–650, New York, New York. IEEE.

Saenko, I. and Kotenko, I. (2012). Design and performance evaluation of improved genetic algorithm for role mining problem. In Stotzka, R., Schiffers, M., and Cotronis, Y., editors, *Proceedings of PDP 2012, Munich, Germany, February 15-17, 2012*, pages 269–274, New York, New York. IEEE.

Saenko, I. and Kotenko, I. (2016a). Reconfiguration of RBAC schemes by genetic algorithms. In Badica, C. and El Fallah Seghrouchni, A. e. a., editors, *Intelligent Distributed Computing X - Proceedings of IDC 2016, Paris, France, October 10-12 2016*, volume 678 of *Studies in Computational Intelligence*, pages 89–98, Cham. Springer.

Saenko, I. and Kotenko, I. (2016b). Using genetic algorithms for design and reconfiguration of RBAC schemes. In *Proceedings of PrAISeECAI 2016, The Hague, Netherlands, August 29-30, 2016*, pages 1–9, New York, New York. ACM.

Vaidya, J., Atluri, V., and Guo, Q. (2007). The role mining problem. In Lotz, V. and Thuraisingham, B., editors, *Proceedings of SACMAT 2007, Sophia Antipolis, France, June 20-22, 2007*, pages 175–184, New York, New York. ACM.

Vaidya, J., Atluri, V., Guo, Q., and Lu, H. (2010a). Role mining in the presence of noise. In Foresti, S. and Jajodia, S., editors, *Data and Applications Security and Privacy XXIV - Proceedings of IFIP 2010, Rome, Italy, June 21-23, 2010.*, volume 6166 of *Lecture Notes in Computer Science*, pages 97–112, Berlin, Heidelberg. Springer.

Vaidya, J., Atluri, V., Warner, J., and Guo, Q. (2010b). Role engineering via prioritized subset enumeration. In *IEEE Transactions on Dependable and Secure Computing*, volume 7, pages 300–314.

Xin, B., Chen, L., Chen, J., Ishibuchi, H., Hirota, K., and Liu, B. (2018). Interactive multiobjective optimization: A review of the state-of-the-art. *IEEE Access*, 6:41256–41279.