IEICE TRANS. INF. & SYST., VOL.E100–D, NO.8 AUGUST 2017

PAPER Special Section on Information and Communication System Security

Towards an Efficient Approximate Solution for the Weighted User Authorization Query Problem

Jianfeng LU[†,††a)], ***Member***, **Zheng WANG**[†], **Dewu XU**[†], **Changbing TANG**[†], ***and*** **Jianmin HAN**[†], ***Nonmembers***

SUMMARY The user authorization query (UAQ) problem determines whether there exists an optimum set of roles to be activated to provide a set of permissions requested by a user. It has been deemed as a key issue for efficiently handling user's access requests in role-based access control (RBAC). Unfortunately, the weight is a value attached to a permission/role representing its importance, should be introduced to UAQ, has been ignored. In this paper, we propose a comprehensive definition of the weighted UAQ (WUAQ) problem with the role-weighted-cardinality and permission-weighted-cardinality constraints. Moreover, we study the computational complexity of different subcases of WUAQ, and show that many instances in each subcase are intractable. In particular, inspired by the idea of the genetic algorithm, we propose an algorithm to approximate solve an intractable subcase of the WUAQ problem. An important observation is that this algorithm can be efficiently modified to handle the other subcases of the WUAQ problem. The experimental results show the advantage of the proposed algorithm, which is especially fit for the case that the computational overhead is even more important than the accuracy in a large-scale RBAC system.

key words: *role-based access control, user authorization query, weight, constraint, genetic algorithm*

1. Introduction

The user authorization query (UAQ) problem for role based access control (RBAC), introduced by Zhang *et al.* [1], is to determine whether there exists an optimum set of roles that should be activated to provide a particular set of permissions requested by a user, which has been the subject of considerable research in recent years, and is widely accepted as a key issue related to efficiently handing users' access requests [2], [3].

The concept of UAQ was originally proposed by Du *et al.* [4] with the name of inter-domain role mapping (IDRM) problem. However, the definition of IDRM is basic and incomplete, as an unique minimal set of roles that exactly covers the requested permissions may not exist. Chen *et al.* [4], [5] redefined the IDRM problem by ensuring two aspects of constraints: (*i*) all of the requested permissions should be available; (*ii*) the principle of least privilege should be observed. Zhang *et al.* [1] generalized IDRM as UAQ, and divided it into three subcases: the exactly match, the availability match, and the least privilege match. Wickramaarachchi *et al.* [6] extended the UAQ problem to a more general definition, in which the number of permissions granted is restricted to both a lower bound and an upper bound. One optimization objective is to prefer a set of roles that has permissions as close to the lower bound as possible, and the other prefers a set of permitted permissions as close to the upper bound as possible. The above specifications for UAQ have considered only the optimization objective for the number of permissions, while the optimization objective for the number of roles is also equally important, which has been largely ignored. Mousavi *et al.* [7] gave a formulation of UAQ as a joint optimization objective for both of the number of roles and the number of permissions. However, they only considered the optimization objective for extra permissions, while the missing permissions is omitted. Hence, Lu *et al.* [2] proposed a comprehensive definition of the UAQ problem by considering the optimization objectives for the number of permissions as well as the number of roles, where the UAQ problem includes two components: the Core-UAQ problem and the Constrained-UAQ problem by introducing the irreducibility, role-cardinality, and permission-cardinality constraints to Core-UAQ. A typical RBAC system may optionally include any Core-UAQ or Constrained-UAQ.

A key limitation of existing works for the specification of UAQ rely on the assumption that there is no difference between permissions, *i.e.*, they did not consider the different nature and importance of each permission, or treated the permissions evenly. However, this is not always true in practice. For example, the permission that *writes to the student achievement* may be more important than the permission *reads of the student achievement*. Unfortunately, the existing works of UAQ simply ignores this difference. Here, we present an example to motivate the new features of the notation about the weight of permission/role to optimize the UAQ problem. Let us assume that the requested permission set is $\{p_2, p_3, p_4\}$, permissions p_1 and p_2 belong to r_1, permissons p_3 and p_4 belong to r_2, permissons p_2, p_4 and p_5 belong to r_3. It is obvious that both $\{r_1, r_2\}$ and $\{r_2, r_3\}$ are the solutions for the available match of UAQ, since each of them has only one extra permissions p_1 and p_5, respectively. However, it may not make any sense for choosing the solution $\{r_1, r_2\}$, if the permission p_1 is more critical than the permission p_5, such as p_1 denotes as the permission that writes to the student achievement, while p_5 represents as the permission that reads to the student achievement. In such

Manuscript received August 19, 2016.
Manuscript revised February 5, 2017.
Manuscript publicized May 18, 2017.

[†]The authors are with School of Mathematics-Physical & Information Engineering, Zhejiang Normal University, Jinhua, Zhejiang, China.
[††]The author is also with State Key Laboratory for Novel Software Technology, Nanjing University, P.R. China.
a) E-mail: lujianfeng@zjnu.edu.cn

DOI: 10.1587/transinf.2016ICP0002

case, the solution $\{r_2, r_3\}$ is a better choice if we consider the weight of the extra permissions. Therefore, the weight of permission/role is a value attached to a permission/role representing its importance [8], [9], which is very important rather than trivial extension that should be introduced to the specification of the UAQ problem, and we name it as the weighted user authorization query (WUAQ) problem.

As discussed in [2], we have studied the computational complexity of the UAQ, and shown that many instances in each subcase with additional constraints are intractable. Since the UAQ problem is only a subcase of WUAQ as it assumes that each permission has the same impact of weight, it is obvious that the WUAQ problem is also intractable. In this work, we continue our research into the computational complexity of the UAQ problem by considering the weights of permissions and roles. A more important issue is how to design efficient algorithms to resolve the intractable instances of the WUAQ problem. In previous, several researchers have proposed exhaustive algorithms to compute a solution for the UAQ problem. Zhang *et al.* [1] proposed a two-step algorithm for the UAQ problem. However, this algorithm may falsely reject some legal success requests. Wickramaarachchi *et al.* [6] introduced two approaches to address the UAQ problem: (*i*) developing algorithm that using the backtracking-based search techniques; (*ii*) reducing the problem to the MAXSAT problem. As pointed out by Armando *et al.* [11], the first approach is exponential-time in design, thus it does not seem to scale to larger RBAC policies. The second approach is unsound, inefficient and offers only limited support for the joint optimization of the number of roles and extra permissions, as pointed out by Mousavi *et al.* [7]. Lu *et al.* [2] proposed an approach to solve the intractable cases of the UAQ problem by employing static pruning, preprocessing and depth-first search based algorithm to reduce its running time. However, the above exhaustive algorithms may not fit with the WUAQ problem due to two major reasons: (*i*) the previous algorithm focus on the numbers of roles and permissions, rather than the weights of them; (*ii*) under the premise of ensuring the accuracy, the computational overhead may be even more important [8]. Thus, an efficient approximate approach is urgently needed for WUAQ.

Briefly, the main contributions of this paper can be summarized as follows.

- We propose a comprehensive definition of the WUAQ problem, by introducing the weight of permission/role to UAQ, and consider the role-weighted-cardinality and permission-weighted-cardinality constraints.
- We study the computational complexity of different subcases of WUAQ problems, and show that many instances in each subcase are intractable.
- We propose an algorithm to approximately solve the intractable cases of the WUAQ problem, an important observation is that this algorithm can be efficiently modified to handle the other subcases of the WUAQ problems.

The rest of this paper is organized as follows. In Sect. 2, we give the formal definition of the WUAQ problem, and study the computational complexity of its variants subcases. Section 3 presents an approximate algorithm to solve the intractable cases of the WUAQ problem. In Sect. 4, we implement the proposed algorithm, and make a comparison with other work. We conclude this paper in Sect. 5.

2. The Weighted User Authorization Query Problem

Ma *et al.* have introduced a formal definition of the weight of permission/role, and given methods of calculating them. For more details, please refer to [8], [9]. In this section, we assume that the weights are given by the system, and introduce two types of constraints: role-weighted-cardinality and permission-weighted-cardinality to the WUAQ problem. Continuing the specification style of UAQ in [2], the WUAQ problem is defined as follows.

Definition 1. ***(The WUAQ Problem)*** *Given a set R of all roles, a set P of all permissions, and a set* P_{req} *of permissions requested by a user u, the WUAQ problem is to identify a role set* $\mathcal{R} \subseteq \mathrm{R}$ *that can be activated by u while may optionally satisfy the following constraints:*

- ***Role-weighted-cardinality:*** *A role-weighted-cardinality constraint is denoted as* $rwc\langle \mathcal{R}, O_r \rangle$*, where* $\mathcal{R} \subseteq R$*,* $O_r \in \{k, \infty^+, \infty^-\}$*, and k is a positive number. We say that* $rwc\langle \mathcal{R}, O_r \rangle$ *is satisfied if and only if the following conditions hold:*
  - $W(\mathcal{R}) \leq k$*, if* $O_r = k$*, where* $W(\mathcal{R})$ *denotes the weight of the role set* $\mathcal{R}$*;*
  - $W(\mathcal{R})$ *is maximized if* $O_r = \infty^+$*;*
  - $W(\mathcal{R})$ *is minimized if* $O_r = \infty^-$*;*
- ***Permission-weighted-cardinality:*** *A permission-weighted-cardinality constraint is denoted as* $pwc\langle \mathcal{R}, O_p \rangle$*, where* $\mathcal{R} \subseteq R$*,* $O_p \in \{t^+, t^-, t^\pm, 0^+, 0^-, 0^\pm\}$*, and t is a positive number. We say that* $\mathcal{R}$ *satisfies* $pwc\langle \mathcal{R}, O_p \rangle$*, if and only if the following conditions hold:*
  - $Perm(\mathcal{R}) \supseteq P_{req}$ *and* $W(Perm(\mathcal{R}) - P_{req}) \leq t$ *if* $O_p = t^+$*, where* $Perm(\mathcal{R})$ *maps a role set* $\mathcal{R}$ *onto a set of all available permissions,* $W(P)$ *denotes the weight of the permission set P;*
  - $Perm(\mathcal{R}) \subseteq P_{req}$ *and* $W(P_{req} - Perm(\mathcal{R})) \leq t$ *if* $O_p = t^-$*;*
  - $W(P_{req} \cup Perm(\mathcal{R}) - P_{req} \cap Perm(\mathcal{R})) \leq t$ *if* $O_p = t^\pm$*;*
  - $Perm(\mathcal{R}) \supseteq P_{req}$ *and for all* $\mathcal{R}' \subseteq R$*,* $Perm(\mathcal{R}') \supseteq P_{req}$ *that* $W(Perm(\mathcal{R}') - P_{req}) \geq W(Perm(\mathcal{R}) - P_{req})$ *if* $O_p = 0^+$*;*
  - $Perm(\mathcal{R}) \subseteq P_{req}$ *and for all* $\mathcal{R}' \subseteq R$*,* $Perm(\mathcal{R}') \subseteq P_{req}$ *that* $W(P_{req} - Perm(\mathcal{R}')) \geq W(P_{req} - Perm(\mathcal{R}))$ *if* $O_p = 0^-$*;*
  - *for all* $\mathcal{R}' \subseteq R$*,* $W(P_{req} \cup Perm(\mathcal{R}') - P_{req} \cap Perm(\mathcal{R}')) \geq W(P_{req} \cup Perm(\mathcal{R}) - P_{req} \cap Perm(\mathcal{R}))$ *if* $O_p = 0^\pm$*;*

A permission-weighted-cardinality constraint specifies an optimization requirement on the weight of the permis-

sions that can be acquired by the requesting user. The parameter t^+ and t^- specify the threshold values that the weight of extra and missing permissions that the system can be able to tolerate, respectively, while $t^\pm$ specifies the threshold values that the weight of both extra and missing permissions. Similarly, the parameters 0^+, 0^- and $0^\pm$ prefer to minimize the weight of extra permissions, missing permissions and the union of them. For instance, when the system can be able to tolerate missing permissions that the total weight of them is 0.5, then t can be set to 0.5 for the case $O_p = t^-$. When the total weight of extra permissions that more than 1.0 may bring the intolerable risk to the system then t can be set to 1.0 for the case $O_p = t^+$. Similarly, if the system want to minimize the total weight of extra and missing permissions, then the case $O_p = 0^\pm$ can satisfy such requirement. In another aspect, a role-weighted-cardinality constraint specifies an optimization requirement on the weight of activated roles. The parameter k specifies the threshold values that the weight of selected roles by the system, while the parameters ∞^+ and ∞^- prefer to maximize and minimize the weight of selected roles, respectively. It may be useful when security constraints make some selected roles to be unavailable, while minimizing the weight of selected roles activated in a user's session may allow an administrator to more efficiently manage the system. For example, when the system can be able to tolerate a set of roles whose total weight is no more than 1.5 that activated in a session, then k can be set to 1.5 for the case $O_r = k$, while the case $O_r = \infty^+$ can satisfy the requirement that the system want to maximize the selected roles.

To specify a subcase of the WUAQ problem, we write it followed by the list of constraints within a pair of braces. For instance, $WUAQ\langle rwc : k + pwc : t^+\rangle$ denotes the subcase that finds the role set $\mathcal{R}$ to satisfy not only $rwc\langle\mathcal{R}, k\rangle$, but also $pwc\langle\mathcal{R}, t^+\rangle$. Note that all the permission-weighted-cardinality constraints can be combined with the role-weighted-cardinality constraints, there may be conflicts between these two types of constraints, we hence simply assign higher priority to the permission-weighted-cardinality constraints than the role-weighted-cardinality to solve such conflicts.

The UAQ problem essentially is a subcase of WUAQ, since it just assumes that each permission and role has the same impact of weight. In this case, determining the computational complexity of the WUAQ problem is a challenge work. In the following, Theorem 1 derives the complexity of different subcases of WUAQ problems.

Theorem 1. *The computational complexity of different subcases of WUAQ problems are as shown in Table 1.*

Proof. The proof for Theorem 1 consists of three parts:

(*i*) We show that $WUAQ\langle rwc : k\rangle$ is NP-complete by proving that it is both NP-hard and in NP. Firstly, it can see that $WUAQ\langle rwc : k\rangle$ is in NP, this is because if one correctly guess a subset $\mathcal{R}$ of R as a solution for $WUAQ\langle rwc : k\rangle$, verifying whether $W(\mathcal{R}) \leq k$, which can be done in polynomial time. Secondly, we prove that $WUAQ\langle rwc : k\rangle$ is NP-hard by reducing the NP-complete subset sum problem [12] to it. The subset sum problem can be depicted as follows: Given a set $A = \{a_i : 1 \leq i \leq n\}$ of positive integers and a positive integer M, the goal is to determine whether there exists a subset of A sum of whose elements equal to a given integer M. The reduction is as follows: Given A and M, construct $WUAQ\langle rwc : k\rangle$ as follows: Let each element in A map to a role weight $W(r) \mid r \in R$ in the problem, and let M equal to k. Next, we construct an RBAC state as follows: For each corresponding role in the element of A, create a single permission p to which the role is covered, and assign the value of $W(r)$ as its weight. The resulting solution $\mathcal{R}$ can be found with respect to $W(\mathcal{R}) \leq k$, if and only if, there exists a subset of A sum of whose elements equal to M.

Table 1 Computational complexities of different subcases of WUAQ problems.

<table>
<tr><th colspan="2" rowspan="2">WUAQ</th><th rowspan="2">None</th><th colspan="3">Role-weighted-cardinality</th></tr>
<tr><th>$O_r = k$</th><th>$O_r = \infty^+$</th><th>$O_r = \infty^-$</th></tr>
<tr><td colspan="2">None</td><td>P</td><td>NP–complete</td><td>P</td><td>P</td></tr>
<tr><td rowspan="6">Permission-weighted-cardinality</td><td>$O_p = t^-$</td><td rowspan="3">NP–complete</td><td rowspan="3">NP^NP</td><td colspan="2" rowspan="3">NP – hard</td></tr>
<tr><td>$O_p = t^+$</td></tr>
<tr><td>$O_p = t^\pm$</td></tr>
<tr><td>$O_p = 0^-$</td><td>P</td><td>NP–hard</td><td>P</td><td>NP–hard</td></tr>
<tr><td>$O_p = 0^+$</td><td colspan="3" rowspan="2">NP – hard</td></tr>
<tr><td>$O_p = 0^\pm$</td></tr>
</table>

(*ii*) We study the complexities of WUAQ problems for the subcases $O_p = t^+$. In fact, the UAQ problem can be regarded as a special case of the WUAQ problem. For example, $UAQ\langle available + pc : t^+\rangle$ in [2] is equal to $WUAQ\langle pwc : t^+\rangle$ when we assume that each permission has the same weight. In this case, the complexity of $WUAQ\langle pwc : t^+\rangle$ is at least as hard as $UAQ\langle available + pc : t^+\rangle$, because any solution for the former problem is also a solution for the later one. It obviously that $WUAQ\langle pwc : t^+\rangle$ is in NP, since verifying whether a given role set is a solution for $WUAQ\langle pwc : t^+\rangle$ can be done in polynomial time. Similarly, the remainder subcases for $O_p = t^+$, and the WUAQ problems for $O_p = t^-$, $O_p = 0^+$ and $O_p = 0^-$ can be derived.

(*iii*) Since the complexities of WUAQ problems for the subcases $O_p = t^\pm$ are at least as hard as both of the cases $O_p = t^+$ and $O_p = t^-$, as any solution for one of the latter two is also a solution for the former one. Similarly, we can derive the complexity of WUAQ problems for the cases $O_p = 0^\pm$.

Other results in Table 1 can be implied from the proved cases. □

3. An Efficient Approximate Algorithm for the Weighted User Authorization Query problem

The fact that WUAQ is intractable, as shown in Theorem 1, means that there exist difficult problem instances that take exponential time in the worst case. However, many instances that will be encountered in practice may still be efficiently solvable. For example, $WUAQ\langle pwc : 0^+\rangle$ is NP-hard as shown in Table 1. In order to solve such intractable subcase, we first propose a binary evolutionary (BE) algorithm to approximate solve it, which is inspired by the idea of the genetic algorithm [13], [14]. Next, we show that our algorithm can be efficiently modified to handle the other subcases of the WUAQ problems.

3.1 The Binary Evolutionary Algorithm

In the BE algorithm, we first generate a population of role sets at random, and evaluate their fitness. Next, we generate a new population based on mutation and crossover with a probability distribution. Finally, the algorithm will stop when iteration times are over the threshold value, and output a solution that approximate solve $WUAQ\langle pwc : 0^+\rangle$. The steps of the BE algorithm are summarized as follows, and the pseudo code is given in Algorithm 1. This algorithm has a time complexity of $O(lmn)$, where l, m, n denote the number of iteration times, the size of population, the number of all available roles, respectively. The main notations used in this paper are shown in Table 2.

(i) **Preprocess:** Remove roles in R that do not have at least one permission in P_{req}.
(ii) **Coding:** Convert each solution of the WUAQ problem to a corresponding chromosome, that use n-bit string $y_1, \cdots, y_n$ where each bit y_i is either 1 or 0 to represent whether the ith role is selected in the solution.
(iii) **Start:** Select a population of m points $x_1, \cdots, x_m$ to represent the roles set at random, and set $l = 0$.
(iv) **Compute fitness:** Compute the fitness of the role set using the evaluation function.
(v) **Replacement:** Sort the m points according to the value of their fitness from large to small, and replace the latter half part by the front half.
(vi) **Regeneration:** If all the points in the first half of the population have the same fitness, set $l = l + 1$, save x_1, and go to step (iii).
(vii) **Mutate:** For each point x_i that $\frac{m}{2} < i \leq m$ in the population and for each bit in x_i, with probability p_{muta}, alter its value.
(viii) **Crossover:** For each y_j in the pair points x_i and $x_{(i+1)}$ from the $x_{\frac{m}{2}}, \cdots, x_m$, with probability p_{cross}, exchange $x_i.y_j$ with $x_{(i+1)}.y_j$.
(ix) **Stop:** Set $l = l + 1$, if l is equal to the threshold value, stop. Otherwise go to step (iv).

Algorithm 1 The BE Algorithm for $WUAQ\langle pwc : 0^+\rangle$

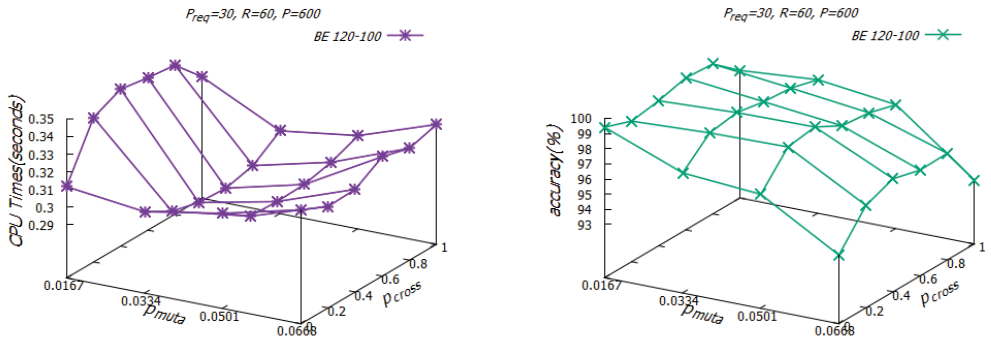
Input: $R, P, RP, P_{req}, W(p), p_{muta}, p_{cross}, T$;
Output: $O.R[n]$
for each $r \in R$ **do**
  if $Perm(r) \cap P_{req} = \emptyset$ **then**
    $R \leftarrow R/\{r\}$
  end if
end for
Rand(E[m]) /* generate a random population*/
while $++l \leq threshold$ **do**
  for each $i \in [0, n)$ **do**
    $E[i].fit \leftarrow p_{size} - w_{ep} - 100w_{mp}$
  end for
  Sort($E[m]$) /*From big to small sort*/
  if $O.fit < E[0].fit$ **then**
    $O \leftarrow E[0]$
  end if
  if $E[0].fit == E[\frac{m}{2} - 1].fit$ **then**
    Rand(E[m])
  end if
  for each $i \leq \frac{m}{2}$ **do**
    $E[\frac{m}{2} + i] \leftarrow E[i]$
  end for
  for each $i > \frac{m}{2}$ **do**
    for each $j \in [0, n)$ **do**
      if $rand() < p_{muta}$ **then**
        $E[i].R[j] \leftarrow \overline{E[i].R[j]}$
      end if
    end for
    for each $i\%2 == 0$ **do**
      for each $j \in [0, n)$ **do**
        if $rand() < p_{cross}$ **then**
          $E[i].R[j] \leftrightarrow E[i + 1].R[j]$
        end if
      end for
    end for
  end for
end while

Table 2 Main notations used in this paper.

Notation	Description
O	The Optimal solution.
$E[m]$	The population of m points.
$E[i].fit$	The fitness of $E[i]$.
$E[i].R[j]$	The jth role in ith point of the population.
p_{cross}	The probability for crossover.
p_{muta}	The probability for mutation.
p_{size}	The size of the total permissions available in the system.
r_{size}	The size of the total roles available in the system.
w_{ep}	The weight of the extra permissions, *i.e.*, $w_{ep} = W(Perm(\mathcal{R}) - P_{req})$.
w_{mp}	The weight of missing permissions, *i.e.*, $w_{mp} = W(P_{req} - Perm(\mathcal{R}))$.
w_{sr}	The weight of selected roles, *i.e.*, $w_{mp} = W(\mathcal{R})$).

3.2 Handling the Other Subcase of the WUAQ Problems

The algorithm described in the previous subsection address a subcase $WUAQ\langle pwc : 0^+\rangle$. An important observation is that this algorithm can handle the other subcases of the



(a) (b)

Fig. 1 The runtime and accuracy for different probability of mutation and crossover.

Table 3 Key differences among the five parameters α, β, γ, δ, ε for various WUAQ problem subcases with role-weighted-cardinality and permission-weighted-cardinality constraints.

WUAQ		None	Role-weighted-cardinality		
			$O_r = k$	$O_r = \infty^-$	$O_r = \infty^+$
None			$0, 1, 0, 0, -\frac{1}{r_{size}}$		$0, 1, 0, 0, \frac{1}{r_{size}}$
Permission-weighted-cardinality	$O_p = t^+$	$1, 0, -1,$	$1, 1, -1,$		$1, 1, -1,$
	$O_p = 0^+$	$-100, 0$	$-100, -\frac{1}{r_{size}}$		$-100, \frac{1}{r_{size}}$
	$O_p = t^-$	$1, 0, -100,$	$1, 1, -100,$		$1, 1 - 100,$
	$O_p = 0^-$	$-1, 0$	$-1, -\frac{1}{r_{size}}$		$-1, \frac{1}{r_{size}}$
	$O_p = t^\pm$	$1, 0, -1,$	$1, 1, -1,$		$1, 1, -1,$
	$O_p = 0^\pm$	$-1, 0$	$-1, -\frac{1}{r_{size}}$		$-1, \frac{1}{r_{size}}$

WUAQ problems by efficiently modifying the evaluation function of fitness. The evaluation function of fitness used in the BE algorithm is defined as follows.

Definition 2. *The evaluation function of fitness is defined as:* $fit = \alpha \times p_{size} + \beta \times r_{size} + \gamma \times w_{ep} + \delta \times w_{mp} + \varepsilon \times w_{sr}$, *where* α, β, γ, δ *and* ε *are parameters used to adjust the relative importance about* p_{size}, r_{size}, w_{ep}, w_{mp} *and* w_{sr}.

Table 3 summarizes the differences among the components used for the different subcases of the WUAQ problem with role weighted-cardinality and permission weighted-cardinality constraints. For example, in order to deal with the subcase $WUAQ\langle pwc : 0^+\rangle$, the evaluation function of fitness is defined as $fit = p_{size} - w_{ep} - 100w_{mp}$, where we set $\alpha = 1$, $\beta = 0$, $\gamma = -1$, $\delta = -100$ and $\varepsilon = 0$. In particularly, $100 \times w_{mp}$ can be regarded as a strict penalty for cases where the chosen roles do not cover all the requested permissions, while $1 \times w_{ep}$ can be regarded as a less penalty for choosing extra permissions. This algorithm will search for the least weight of extra permissions, which is a solution for $WUAQ\langle pwc : 0^+\rangle$.

4. Experimental Results

In order to show the advantage of the proposed BE algorithm, we have implemented it and performed several experiments using randomly generated instances. We make a comparison of our BE algorithm with the Depth-First Search (DFS) algorithm in [2] for solving the subcase $WUAQ\langle pwc : 0^+\rangle$ for two major reasons: Firstly, the comparison of the DFS with the Backtracking-Based Search (BBS) algorithm proposed in [6] has shown the effectiveness of the DFS algorithm. Secondly, both BE and DFS algorithm can be efficiently modified to handle the other subcases of the WUAQ/UAQ problems. The implementation of both BE and DFS algorithms were written in C. All the experiments have been carried out on a standard desktop PC with a Intel(R) Core(TM) i7-4790 CPU running at 3.60GHz, and with DDR3 16GB 1600MHz memory, running Microsoft windows 7 Ultimate Editions. For each instance, 10 randomly generated test cases are run, the averages of the test results are used to generate the graphs.

4.1 Effectiveness of Mutation and Crossover

In order to verify the accuracy of the solutions obtained. We first implemented the DFS algorithm to generate an optimal solution $\mathcal{R}_{DFS}$ for reasonable size problems. We next ran the BE algorithm to get a role set $\mathcal{R}_{BE}$, and computed the accuracy of the BE algorithm, which was defined as follow.

Definition 3. *The accuracy of the BE algorithm is defined as:* $\left(1 - \frac{W(Perm(\mathcal{R}_{BE})) - W(Perm(\mathcal{R}_{DFS}))}{W(Perm(\mathcal{R}_{DFS}))}\right) \times 100\%$.

Figure 1 shows the average CPU times and accuracy under different probability of crossover and mutation for the test case: $P_{req} = 30$, $R = 60$, $P = 600$, the size of population $m = 120$, and the number of iteration $l = 100$. The x-axis denotes the probability of mutation which we fix its value as $\frac{1}{r_{size}}$, $\frac{2}{r_{size}}$, $\frac{3}{r_{size}}$ and $\frac{4}{r_{size}}$ respectively. It can be clearly seen from Fig. 1 (a) that the average CPU times is least when we choose the parameter $p_{muta} = \frac{2}{r_{size}}$ for the fixed p_{cross}, and the average CPU times increases with the maximal p_{cross} for

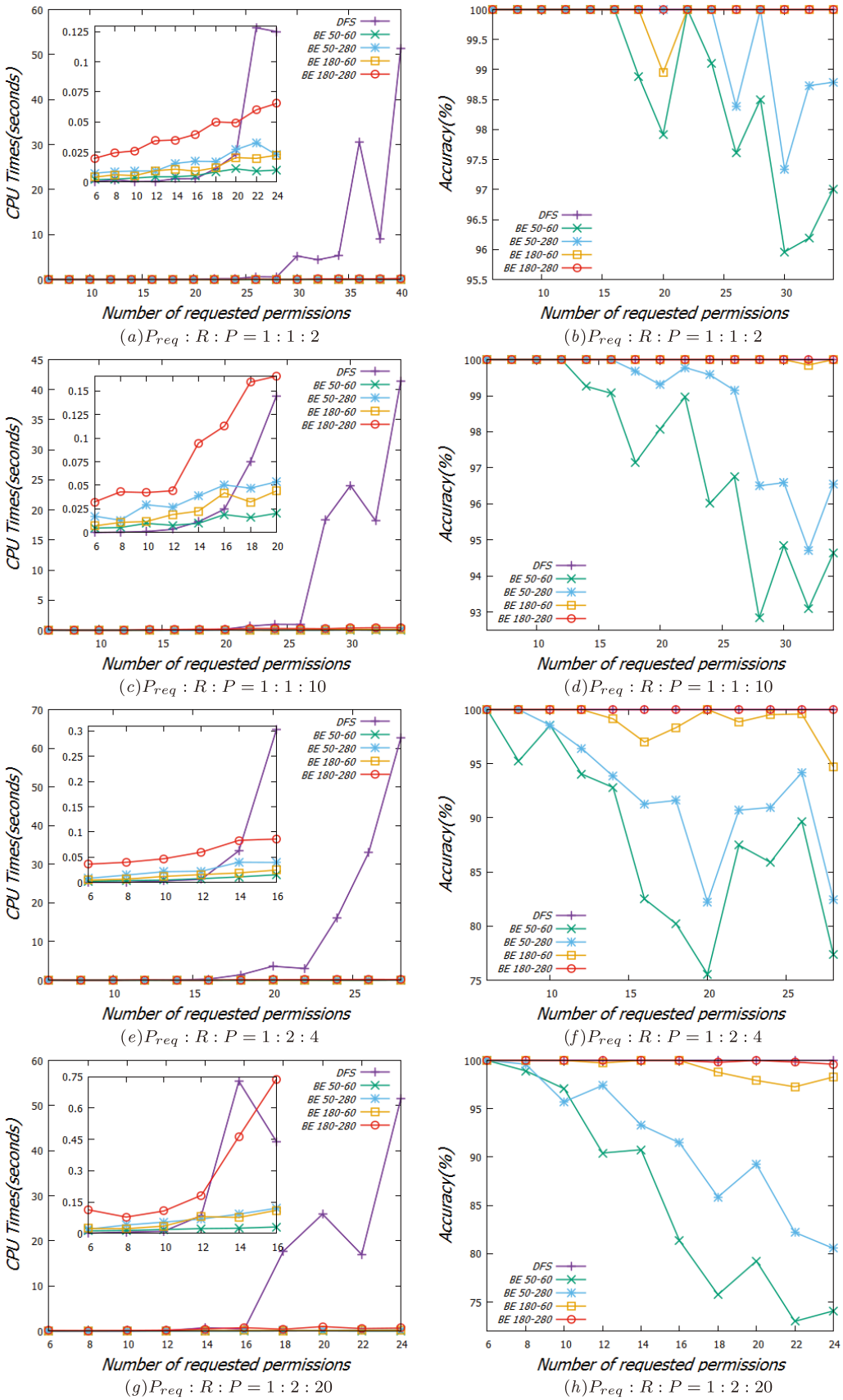


Fig. 2 Running time and accuracy for Binary Evolutionary (BE) algorithm and Depth First Search (DFS) algorithm.

the fixed p_{muta}. This is reasonable because the BE algorithm depends on the generated data and the size of the role set R, hence the less p_{cross} will save the CPU times. But the impact of p_{muta} for CPU times is less. As shown in Fig. 1 (b), the average accuracy increases with the maximal p_{cross} for the fixed p_{muta}. This is because it is easy to find an optimal solution if we assign a big value for p_{cross}. However, we can't get the optimal solution when the value of p_{cross} is either maximum or minimum. Together with the observation, we choose the parameters $p_{muta} = \frac{2}{r_{size}}$ and $p_{cross} = 0.6$ for the remainder experiments.

4.2 Comparison of the BE Algorithm with the DFS Algorithm

Figure 2 shows the results of running the experiments for the four test case (1) $P_{req} : R : P = 1 : 1 : 2$; (2) $P_{req} : R : P = 1 : 1 : 10$; (3) $P_{req} : R : P = 1 : 2 : 4$; (4) $P_{req} : R : P = 1 : 2 : 20$. In particularly, we generate four different instances: *BE* 50-60, *BE* 50-280, *BE* 180-60, *BE* 180-280, the first parameter denotes the size of the population, and the second parameter denotes the number of iteration in this algorithm. The runtime and accuracy of these two algorithms depend on the total number of the requested permissions P_{req}, available roles R and available permissions P.

In Figs. 2 (a), (c), (e) and (g), both of these two algorithms produce comparable results when the number of requested permissions is small. However, as the number of requested permissions increases, the overall CPU time taken increases exponentially, this makes the DFS algorithm impractical for implementation in dynamic systems. However, the BE algorithm with different instances take a few seconds, even for a larger number of roles, permissions and requested permissions. The reason is that the BE algorithm will stop when iteration times are over the threshold value. It is worth noting that the BE algorithm turns out to be more effective when both of the population m and iteration l are small. Such as *BE* 50-60 always has the least CPU times, and *BE* 180-280 always has the largest CPU times compare with the other two subcases of BE. However, the accuracy of BE is close to the DFS algorithm when we choose large number of m and l. As the number of requested permissions increases, the accuracy of the BE algorithm decreases. For example, the accuracy of *BE* 50-60 is less than 75% for a bad instance, as shown in Fig. 2 (h).

Consequently, for the case that the accuracy is not very critical, we can make the accuracy of the BE algorithm in an acceptable extent by enlarging the value of population size and iteration. The BE algorithm is able to efficiency approximate solve the WUAQ problem even though the number of permissions in a larger scale RBAC.

5. Conclusion

In this paper, we introduced the concept of permission/role weight, and gave a formal definition for the WUAQ problem by considering two types of constraints: role-weighted-cardinality and permission-weighted-cardinality. In fact, WUAQ is a more comprehensive definition that includes the UAQ problem. Furthermore, we studied the computational complexity analysis of various subcases of the WUAQ, and showed that most instances of WUAQ problem are intractable. In particular, we proposed a BE algorithm to efficient approximate solve an instance of WUAQ, and showed how it can be efficiently modified to handle the other subcases. The comparison between the BE algorithm and DFS algorithm shown the efficiency and accuracy of the proposed BE algorithm. This algorithm is especially fit for the case that the computational overhead is even more important than the accuracy.

Acknowledgements

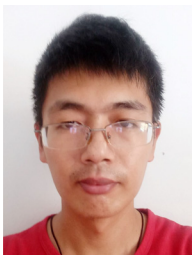
This work is supported by National Natural Science Foundation of China under Grant 61402418, 61503342, 61672468, 61602418, Social development project of Zhejiang provincial public technology research under Grant 2017C33054, MOE (Ministry of Education in China) Project of Humanity and Social Science under Grant 12YJCZH142, Zhejiang Provincial Natural Science Foundation of China under Grant LY13F020017, LY15F020013, LY16F030002.

References

[1] Y. Zhang and J.B.D. Joshi, "UAQ: A framework for user authorization query processing in RBAC extended with hybrid hierarchy and constraints," Proc. 13th ACM Symposium on Access Control Models and Technologies, New York, NY, USA, pp.83–92, 2008.
[2] J. Lu, J.B.D. Joshi, L. Jin, and Y. Liu, "Towards complexity analysis of user authorization query problem in RBAC," Computers & Security, vol.48, pp.116–130, 2015.
[3] J.-F. Lu, J.-M. Han, W. Chen, and J.-W. Hu, "Safety and availability checking for user authorization queries in RBAC," International Journal of Computational Intelligence Systems, vol.5, no.5, pp.860–867, 2012.
[4] S. Du and J.B.D. Joshi, "Supporting authorization query and inter-domain role mapping in presence of hybrid role hierarchy," Proc. 11th ACM Symposium on Access Control Models and Technologies, Lake Tahoe, California, USA, pp.228–236, 2006.
[5] L. Chen and J. Crampton, "Inter-domain role mapping and least privilege," Proc. 12th ACM Symposium on Access Control Models and Technologies, Sophia Antipolis, France, pp.157–162, 2007.
[6] G.T. Wickramaarachchi, W.H. Qardaji, and N. Li, "An efficient framework for user authorization queries in RBAC systems," Proc. 14th ACM Symposium on Access Control Models and Technologies, Stresa, Italy, pp.23–32, 2009.
[7] N. Mousavi and M.V. Tripunitara, "Mitigating the intractability of the user authorization query problem in role-based access control (RBAC)," Proc. 6th International Conference on Network and System Security, Fujian, China, pp.516–529, 2012.
[8] X. Ma, R. Li, Z. Lu, J. Lu, and M. Dong, "Specifying and enforcing the principle of least privilege in role-based access control," Concurrency and Computation: Practice and Experience, vol.23, no.12, pp.1313–1331, 2011.
[9] X. Ma, R. Li, and Z. Lu, "Role mining based on weights," Proc. 15th ACM Symposium on Access Control Models and Technologies (SACMAT 2010), Pittsburgh, PA, USA, pp.65–74, 2010.
[10] L. Chen and J. Crampton, "Set cover problems in role-based access

control," Proc. 14th European Symposium on Research in Computer Security, Saint Malo, France, pp.689–704, 2009.
[11] A. Armando, S. Ranise, F. Turkmen, and B. Crispo, "Efficient run-time solving of RBAC user authorization queries: Pushing the envelope," Proc. ACM Conference on Data and Application Security and Privacy, San Antonio, Texas, USA, pp.241–248, 2012.
[12] S. Arora and B. Barak, Computational Complexity: A Modern Approach, Cambridge University Press, 2009.
[13] I. Saenko and I. Kotenko, "Genetic algorithms for role mining problem," Proc. 19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2011), Ayia Napa, Cyprus, pp.646–650, 2011.
[14] I. Saenko and I. Kotenko, "Design and performance evaluation of improved genetic algorithm for role mining problem," Proc. 20th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2012), Garching near Munich, Germany, pp.269–274, 2012.

Jianfeng Lu is an associate professor in the School of Mathematics-Physical and Information Engineering at Zhejiang Normal University. He received his B.S. degree in the School of Computer Science and Technology at Wuhan University of Science and Technology in 2005, and the Ph.D. degree in the School of Computer Science and Technology at Huazhong University of Science and Technology in 2010. In 2013, he was a visiting researcher with the University of Pittsburgh, Pittsburgh, USA. His currently research interests include access control and game theory.



Zheng Wang is currently a graduate student in the School of Mathematics-Physical and Information Engineering at Zhejiang Normal University. He is expected to graduate in June 2017. His research interests include access control and optimization algorithm.

Dewu Xu is an associate professor in the School of Economics and Management at Zhejiang Normal University. He received his M.S. degree in School of Information Science and Technology at East China Normal University in 2005. His research interests include cryptography and distributed system security.

Changbing Tang is an assistant professor in the School of Mathematics-Physical and Information Engineering at the Zhejiang Normal University. He received his B.S. and M.S. degrees in the mathematics and applied mathematics from Zhejiang Normal University, in 2004 and 2007, respectively, and the Ph.D. degree in the Department of Electronic Engineering at Fudan University in 2014. His research interests include game theory and its applications.

Jianmin Han is a professor in the School of Mathematics-Physical and Information Engineering at Zhejiang Normal University. He received his B.S. degree in the department of Computer Science and Technology at DaQing Petroleum institute in 1992, and the Ph.D. degree in the department of Computer Science and Technology at East China University of Science and Technology in 2009. His research interests include privacy preservation and game theory.