Valued Authorization Policy Existence Problem: Theory and Experiments

JASON CRAMPTON, Royal Holloway, University of London, United Kingdom
EDUARD EIBEN, Royal Holloway, University of London, United Kingdom
GREGORY GUTIN, Royal Holloway, University of London, United Kingdom
DANIEL KARAPETYAN, University of Nottingham, United Kingdom
DIPTAPRIYO MAJUMDAR, Royal Holloway, University of London, United Kingdom

Recent work has shown that many problems of satisfiability and resiliency in workflows may be viewed as special cases of the authorization policy existence problem (APEP), which returns an authorization policy if one exists and “No” otherwise. However, in many practical settings it would be more useful to obtain a “least bad” policy than just a “No”, where “least bad” is characterized by some numerical value indicating the extent to which the policy violates the base authorization relation and constraints. Accordingly, we introduce the Valued APEP, which returns an authorization policy of minimum weight, where the (non-negative) weight is determined by the constraints violated by the returned solution.

We then establish a number of results concerning the parameterized complexity of Valued APEP. We prove that the problem is fixed-parameter tractable (FPT) if the set of constraints satisfies two restrictions, but is intractable if only one of these restrictions holds. (Most constraints known to be of practical use satisfy both restrictions.)

We also introduce a new type of resiliency for workflow satisfiability problem, show how it can be addressed using Valued APEP and use this to build a set of benchmark instances for Valued APEP. Following a set of computational experiments with two mixed integer programming (MIP) formulations, we demonstrate that the Valued APEP formulation based on the user profile concept has FPT-like running time and usually significantly outperforms a naive formulation.

CCS Concepts: • **Security and privacy → Formal methods and theory of security**; • **Theory of computation → Fixed parameter tractability**.

Additional Key Words and Phrases: access control, workflow satisfiability, Authorization Policy Existence Problem, resiliency problems

ACM Reference Format:
Jason Crampton, Eduard Eiben, Gregory Gutin, Daniel Karapetyan, and Diptapriyo Majumdar. 2018. Valued Authorization Policy Existence Problem: Theory and Experiments. In *Woodstock ’18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 32 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Access control is a fundamental aspect of the security of any multi-user computing system. Access control requirements are typically enforced by specifying an authorization policy and implementing a system to enforce the policy. Such a policy identifies which interactions between users and resources are to be allowed (and denied) by the access control system.

Over the years, authorization policies have become more complex, not least because of the introduction of constraints – often motivated by business requirements such as “Chinese walls” – which further refine an authorization policy. A separation-of-duty constraint (also known as the "two-man rule" or "four-eyes policy") may, for example, require that no single user is authorized for some particularly sensitive group of resources. Such a constraint is typically used to prevent misuse of the system by a single user.

The use of authorization policies and constraints, by design, limits which users may access resources. Nevertheless, the ability to perform one's duties will usually require access to particular resources, and overly prescriptive policies and constraints may mean that some resources are unavailable to users that need access to them. In other words, there may be some conflict between authorization policies and operational demands: a policy that is insufficiently restrictive may suit operational requirements but lead to security violations; conversely, too restrictive a policy may compromise an organization's ability to meet its business objectives.

Recent work on workflow satisfiability and access control resiliency recognized the importance of being able to determine whether or not security policies prevent an organization from achieving its objectives [10, 11, 24, 26, 30]. Bergé *et al.* introduced the Authorization Policy Existence Problem (APEP) [2], which generalizes many of the existing satisfiability and resiliency problems in access control. Informally, the APEP seeks to find an authorization policy, subject to restrictions on individual authorizations (defined by a base authorization relation) and restrictions on collective authorizations (defined by a set of authorization constraints).

APEP may be viewed as a decision or search problem. An algorithm to solve either version of the problem returns "no" if no authorization policy exists, given the base authorization relation and the constraints that form part of the input to the instance. Such a response is not particularly useful in practice: from an operational perspective, an administrator would presumably find it more useful if an algorithm to solve APEP returned some policy, even if that policy could lead to security violations, provided the risk of deploying that policy could be quantified in some way.

Hence, in this paper, we introduce a generalization of APEP, which we call Valued APEP, where every policy is associated with a non-negative weight. A solution to Valued APEP is a policy of minimum weight; a policy of zero weight satisfies the base authorization relation and all the constraints.

We establish the complexity of Valued APEP for certain types of constraints, using multi-variate complexity analysis. We prove that APEP (and hence Valued APEP) is fixed-parameter intractable, even if all the constraints are user-independent, a class of constraints for which the Workflow Satisfiability Problem (WSP) – a special case of APEP – is fixed-parameter tractable. However, we subsequently show that Valued APEP is fixed-parameter tractable when all weighted constraints are user-independent and the set of constraints is t-weight-bounded (t-wbounded). Informally, the identities of the users are irrelevant to the solution and there exists a solution (policy) containing no more than t authorizations. We show that sets of user-independent constraints that contain only particular kinds of widely used constraints are t-wbounded. Bergé *et al.* [2] introduced and used a notion of a bounded constraint. Bounded and wbounded constraints have some similarities, but wbounded constraints are more refined and allow for more precise complexity analysis. In particular, the notion of a bounded constraint cannot be used for Valued APEP and we are able to derive improved complexity results for APEP using wbounded constraints.

A significant innovation of the paper is to introduce the notion of a user profile for a weighted constraint. Counting user profiles provides a powerful means of analyzing the complexity of (Valued) APEP, somewhat analogous to the use of patterns in the analysis of workflow satisfiability problems. This enables us to (i) derive the complexity of Valued APEP when all constraints are t-wbounded and user-independent, (ii) establish the complexity of Valued APEP for the most common types of user-independent constraints, and (iii) improve on existing results for the complexity of APEP obtained by Bergé *et al.* [2]. We also prove that our result for the complexity of APEP with t-wbounded, user-independent constraints cannot be improved, unless a well-known and widely accepted hypothesis in parameterized complexity theory is false. Finally, we show that certain sub-classes of Valued APEP can be reduced to the Valued Workflow Satisfiability Problem (Valued WSP) [12] with user-independent constraints, thereby establishing that these sub-classes are fixed-parameter tractable.

For the first time in the APEP literature, we conduct computational experiments based on an application of Valued APEP in WSP. Specifically, we introduce a concept of τ-resiliency in WSP which seeks a solution that is resilient to deleting up to τ arbitrary users. We build a set of Valued APEP benchmark instances that address τ-resiliency in WSP and use it in computational experiments. To solve Valued APEP, we use two mixed integer programming formulations. One is a 'naive' formulation of the problem whereas the other one exploits the user profile concept. We demonstrate that the formulation based on the user profile concept has FPT-like solution times and usually outperforms the naive formulation by a large margin. We also analyse and discuss the properties of τ-resiliency.

In the next section, we summarize relevant background material. We introduce the Valued APEP in Section 3 and define weighted user-independent constraints. We also show that Valued WSP is a special case of Valued APEP and describe particular types of weighted user-independent constraints for APEP. In Section 4, we introduce the notion of a t-wbounded constraint and establish the complexity of Valued APEP when all constraints are t-wbounded. We prove the problem is intractable for arbitrary sets of t-wbounded constraints or user-independent constraints, but fixed-parameter tractable for t-wbounded, user-independent constraints. In the following two sections, we establish the complexity of other sub-classes of Valued APEP. In Section 7 we explain how APEP can be used to address questions of resiliency in workflows. In the following two sections we introduce two MIP formulations for Valued APEP and test these formulations using the resiliency questions introduced in Section 7. In Section 10 we discuss how our contributions improve and extend related work. We conclude the paper with a summary of our contributions and some ideas for future work in Section 11.

2 BACKGROUND

APEP is defined in the context of a set of *users* U, a set of *resources* R, a *base authorization relation* $\hat{A} \subseteq U \times R$, and a set of *constraints* C. Informally, APEP asks whether it is possible to find an authorization relation A that satisfies all the constraints and is a subset of $\hat{A}$.

For an arbitrary authorization relation $A \subseteq U \times R$ and an arbitrary resource $r \in R$, we write $A(r)$ to denote the set of users authorized for resource r by A; more formally, $A(r) = \{u \in U \mid (u, r) \in A\}$. For a subset $T \subseteq R$, we define the set of users authorized for some resource in T to be $A(T) = \bigcup_{r \in T} A(r)$. For a user u, $A(u) = \{r \in R \mid (u, r) \in A\}$; and for $V \subseteq U$, $A(V) = \{r \in R \mid (u, r) \in A, u \in V\}$.

An authorization relation $A \subseteq U \times R$ is

- *authorized* (with respect to $\hat{A}$) if $A \subseteq \hat{A}$,
- *complete* if for all $r \in R$, $A(r) \neq \emptyset$,
- *eligible* (with respect to C) if it satisfies all $c \in C$,
- *valid* (with respect to $\hat{A}$ and C) if A is authorized, complete, and eligible.

An instance of APEP is *satisfiable* if it admits a valid authorization relation A.

2.1 APEP constraints

In general, there are no restrictions on the constraints that can appear in an APEP instance, although the use of arbitrary constraints has a significant impact on the computational complexity of APEP (see Section 2.3). Accordingly, Bergé *et* *al.* [2] defined several standard types of constraints for APEP, summarized in Table 1, generalizing existing constraints in the access control literature.

Table 1. Standard APEP constraints: $r, r' \in R$, $t \in \mathbb{N}$

Description	Notation	Satisfaction criterion	Constraint family
Universal binding-of-duty	$(r, r', \leftrightarrow, \forall)$	$A(r) = A(r')$	$\mathrm{BoD_U}$
Universal separation-of-duty	$(r, r', \updownarrow, \forall)$	$A(r) \cap A(r') = \emptyset$	$\mathrm{SoD_U}$
Existential binding-of-duty	$(r, r', \leftrightarrow, \exists)$	$A(r) \cap A(r') \neq \emptyset$	$\mathrm{BoD_E}$
Existential separation-of-duty	$(r, r', \updownarrow, \exists)$	$A(r) \neq A(r')$	$\mathrm{SoD_E}$
Cardinality-Upper-Bound	$(r, \leq, t)$	$\lvert A(r)\rvert \leq t$	$\mathrm{Card_{UB}}$
Cardinality-Lower-Bound	$(r, \geq, t)$	$\lvert A(r)\rvert \geq t$	$\mathrm{Card_{LB}}$

2.2 WSP as a special case of APEP

Consider an instance of APEP which contains any of the constraints defined in Section 2.1, and includes the set of cardinality constraints $\{(r, \leq, 1) \mid r \in R\}$. Any solution A to such an APEP instance requires $|A(r)| = 1$ for all $r \in R$ (since completeness requires $|A(r)| > 0$). Thus A may be regarded as a function $A : R \to U$. Since $|A(r)| = 1$, there is no distinction between existential and universal constraints (whether they are separation-of-duty or binding-of-duty): specifically, A satisfies the constraint $(r, r', \circ, \exists)$ iff A satisfies $(r, r', \circ, \forall)$ (for $\circ \in \{\updownarrow, \leftrightarrow\}$).

In other words, an APEP instance of this form is equivalent to an instance of WSP [11, 30], with separation-of-duty, binding-of-duty and cardinality constraints: resources correspond to workflow steps, the base authorization relation to the authorization policy, and an APEP solution to a plan. Accordingly, strong connections exist between APEP and WSP, not least because certain instances of APEP can be reduced to WSP [2]. In WSP, the set of resources is the set of *steps*, denoted by S.

2.3 Complexity of WSP and APEP

In the context of WSP, the authorization policy (the base authorization relation in APEP) specifies which users are authorized for which steps in the workflow. A solution to WSP is a plan π that assigns a single user to each step in the workflow. In general, WSP is NP-complete [30].

Let $k = |S|$ and $n = |U|$. Then there are n^k plans, and the validity of each plan can be established in polynomial time (in the size of the input). Thus WSP can be solved in polynomial time if k is constant. It is easy to establish that APEP is harder than WSP in general.

Proposition 2.1. APEP *is* NP*-complete even when there is a single resource.*

We provide a polynomial time redution from Monotone 1-in-3 SAT [29] problem. We formally state the problem definition.

> Monotone 1-in-3 SAT
> **Input:** A 3-CNF formula ϕ such that no literal is a negated variable.
> **Question:** Does ϕ have a satisfying assignment that assigns True to only one literal from every clause?

The proof uses a simple reduction from Monotone 1-in-3 SAT [29] to an instance of APEP in which there is a single resource r: the set of variables corresponds to the set of users; $(x, r) \in A$ corresponds to assigning the value True to variable x; and every clause corresponds to a constraint comprising three "users", which is satisfied provided exactly one user is assigned to the resource.

Wang and Li [30] introduced parameterization[1] of WSP by parameter k. This parameterization is natural because for many practical instances of WSP, $k = |S| \ll n = |U|$ and k is relatively small. Wang and Li proved that WSP is intractable, even from the parameterized point of view. However, Wang and Li proved that WSP becomes computationally tractable from the parameterized point of view (i.e., fixed-parameter tractable) when the constraints are restricted to some generalizations of binary separation-of-duty (SoD) and binding-of-duty (BoD) constraints.

Similarly, for APEP, we denote $k = |R|$ and $n = |U|$. In the rest of the paper, we assume that k is relatively small and thus consider it as the parameter. While the assumption that k is small is not necessarily correct in some applications, our approach is useful where k is indeed small, for example in special cases such as WSP. Also, there are situations where strict controls are placed on the utilization of and access to (some small subset of system) resources by users.

2.4 User-independent constraints

Wang and Li's result has been extended to the much larger family of user-independent constraints, which includes the aforementioned SoD and BoD constraints and most other constraints that arise in practice [7, 22]. Informally, a constraint is called user-independent if its satisfaction does not depend on the identities of the users assigned to steps. (For example, it is sufficient to assign steps in a separation of duty constraint to different users in order to satisfy the constraint.)

The concept of a user-independent constraint for WSP can be extended formally to the APEP setting in the following way [2]. Let $\sigma : U \to U$ be a permutation on the user set. Then, given an authorization relation $A \subseteq U \times R$, we write $\sigma(A) = \{(\sigma(u), r) | (u, r) \in A\}$. A constraint c is said to be *user-independent* if for every authorization relation A that satisfies c and every permutation $\sigma : U \to U$, $\sigma(A)$ also satisfies c. It is not hard to see that the sets of constraints defined in Section 2.1 are user-independent [2], since their satisfaction is independent of the specific users that belong to $A(r)$ and $A(r')$.

Bergé *et al.* established a number of FPT results for APEP (restricted to t-bounded, user-independent constraints). We introduce a definition of user-independence and t-boundedness for weighted constraints in Sections 3 and 4, respectively, and show that we can improve on existing complexity results.

2.5 Parameterized complexity

An instance of a parameterized problem Π is a pair (I, κ) where I is the *main part* and κ is the *parameter*; the latter is usually a non-negative integer. A parameterized problem is *fixed-parameter tractable* (FPT) if there exists a computable function f such that any instance (I, κ) can be solved in time $O(f(\kappa)|I|^c)$, where $|I|$ denotes the size of I and c is an absolute constant. The class of all fixed-parameter tractable decision problems is called FPT and algorithms which run in the time specified above are called FPT algorithms. As in other literature on FPT algorithms, we will often omit the polynomial factor in $O(f(\kappa)|I|^c)$ and write $O^*(f(\kappa))$ instead.

Consider two parameterized problems Π and Π'. We say that Π has a *parameterized reduction* to Π' if there are functions g and h from $\mathbb{N}$ to $\mathbb{N}$ and a function $(I, \kappa) \mapsto (I', \kappa')$ from Π to Π' such that

- there is an algorithm of running time $h(\kappa) \cdot (|I| + \kappa)^{O(1)}$ which for input (I, κ) outputs (I', κ'), where $\kappa' \leq g(\kappa)$; and

[1]We provide a brief introduction to parameterized complexity in Section 2.5.

- (I, κ) is a yes-instance of Π if and only if (I', κ') is a yes-instance of Π'.

While FPT is a parameterized complexity analog of P in classical complexity theory, there are many parameterized hardness classes, forming a nested sequence of which FPT is the first member: FPT$\subseteq$ W[1]$\subseteq$ W[2] $\subseteq \ldots$. The *Exponential Time Hypothesis* (ETH) is a well-known and plausible conjecture that there is no algorithm solving 3-CNF Satisfiability in time $2^{o(n)}$, where n is the number of variables [19]. It is well known that if the ETH holds then FPT $\neq$ W[1]. Hence, W[1] is generally viewed as a parameterized intractability class, which is an analog of NP in classical complexity.

A well-known example of a W[1]-complete problem is the Clique problem parameterized by κ: given a graph G and a natural number κ, decide whether G has a complete subgraph on κ vertices. A well-known example of a W[2]-complete problem is the Dominating Set problem parameterized by κ: given a graph $G = (V, E)$ and a natural number κ, decide whether G has a set S of κ vertices such that every vertex in $V \setminus S$ is adjacent to some vertex in S. Thus, every W[1]-hard problem Π_1 is at least as hard as Clique (i.e., Clique has a parameterized reduction to Π_1); similarly, every W[2]-hard problem Π_2 is at least as hard as Dominating Set.

More information on parameterized algorithms and complexity can be found in recent books [14, 17].

3 VALUED APEP

As we noted in the introduction, we believe that it is more valuable, in practice, for APEP to return some authorization relation, even if that relation is not valid (in the sense defined in Section 2). Clearly, the authorization relation that is returned must be the best one, in some appropriate sense. Inspired by Valued WSP, we introduce Valued APEP, where every authorization relation is associated with a "cost" (more formally, a *weight*) and the solution to a Valued APEP instance is an authorization relation of minimum weight.

3.1 Problem definition

We first introduce the notions of a *weighted constraint* and a *weighted user authorization function*. Let $A \subseteq U \times R$ be an authorization relation. A weighted constraint c is defined by a function $w_c : 2^{U \times R} \to \mathbb{N}$ such that $w_c(A) = 0$ if and only if A satisfies the constraint. Hence, we will use interchangeably c and w_c as a notation for c. By definition, $w_c(A) > 0$ if the constraint is violated. The intuition is that $w_c(A)$ represents the cost incurred by A, in terms of constraint violation. For example, a weighted constraint w_c such that $w_c(A) = 0$ iff $A(r) \cap A(r') = \emptyset$ and $w_c(A)$ increases monotonically with the size of $A(r) \cap A(r')$ encodes the usual APEP constraint $(r, r', \updownarrow, \forall)$. (We describe other weighted constraints in Section 3.2.)

A weighted user authorization function $\omega : U \times 2^R \to \mathbb{N}$ has the following properties:

$$\omega(u, T) = 0 \quad \text{if } u \text{ is authorized for each resource in } T \tag{1}$$

$$T' \subseteq T \quad \text{implies} \quad \omega(u, T') \leq \omega(u, T). \tag{2}$$

Then $\omega(u, T) > 0$ if u is not authorized for some resource in T and, vacuously, we have $\omega(u, \emptyset) = 0$ for all $u \in U$. The weighted user authorization function is used to represent the cost of assigning unauthorized users to resources.

Then we define the weighted authorization function $\Omega : 2^{U\times R} \to \mathbb{N}$, weighted constraint function $w_C : 2^{U\times R} \to \mathbb{N}$, and weight function $w : 2^{U\times R} \to \mathbb{N}$ as follows:

$$\Omega(A) = \sum_{u\in U} \omega(u, A(u)), \tag{3}$$

$$w_C(A) = \sum_{c\in C} w_c(A), \tag{4}$$

$$w(A) = \Omega(A) + w_C(A). \tag{5}$$

A relation A is *optimal* if $w(A) \leq w(A')$ for all $A' \subseteq U \times R$. We now formally define Valued APEP.

Valued APEP
Input: A set of resources R, a set of users U, a set of weighted constraints C, a weighted user authorization function ω
Parameter: $\lvert R\rvert = k$
Output: A complete, optimal authorization relation

Remark 3.1. *A base authorization relation $\hat{A}$ is implicitly defined in a* Valued APEP *instance: specifically, $(u, r) \in \hat{A}$ iff $\omega(u, r) = 0$. An instance of* Valued APEP *is defined by a tuple (R, U, C, ω), where C is a set of weighted constraints; we may, when convenient, refer to $\hat{A}$, as defined by ω.*

3.2 Valued APEP constraints

We now provide some examples of weighted constraints, extending the examples introduced in Section 2.1. First, let $f_c : \mathbb{Z} \to \mathbb{N}$ be a monotonically increasing function (i.e., $f_c(z) \leq f_c(z+1)$ for all $z \in \mathbb{Z}$), where $f_c(z) = 0$ iff $z \leq 0$, and let ℓ_c be some constant. Define $\mathrm{maxdiff}(A, r, r')$ to be $\max\{|A(r) \setminus A(r')|, |A(r') \setminus A(r)|\}$. Then the equations below demonstrate how an unweighted APEP constraint c may be extended to a weighted constraint w_c using f_c.

Unweighted	**Weighted**	
$(r, \leq, t)$	$w_c(A) = f_c(\lvert A(r)\rvert - t)$	(6)
$(r, \geq, t)$	$w_c(A) = f_c(t - \lvert A(r)\rvert)$	(7)
$(r, r', \updownarrow, \forall)$	$w_c(A) = f_c(\lvert A(r) \cap A(r')\rvert)$	(8)
$(r, r', \leftrightarrow, \forall)$	$w_c(A) = f_c(\mathrm{maxdiff}(A, r, r'))$	(9)
$(r, r', \updownarrow, \exists)$	$w_c(A) = \begin{cases} 0 & \text{if } A(r) \neq A(r'), \\ \ell_c & \text{otherwise.} \end{cases}$	(10)
$(r, r', \leftrightarrow, \exists)$	$w_c(A) = \begin{cases} 0 & \text{if } A(r) \cap A(r') \neq \emptyset, \\ \ell_c & \text{otherwise.} \end{cases}$	(11)

For example, the weighted cardinality constraint (6) evaluates to 0 if A assigns no more than t users to r, and some non-zero value determined by f_c and $|A(r)|$ otherwise. Note that we may write $f_c(x) = \max\{0, g_c(x)\}$ for some $g_c(x)$. The specific choice of function g_c and the constant ℓ_c will vary, depending on the particular application and particular constraint that is being encoded. For example, one may choose the following functions g_c: $g_c(x) = \lceil a \log_2(1 + x)\rceil$, $g_c(x) = ax$, $g_c(x) = ax^2$ for some positive integer a. For a given constraint c, the choice among the three functions above and the value of a may depend on the degree of importance for c to be satisfied. For notational convenience, we may refer to binding-of-duty and separation-of-duty constraints of the form $(r, r', \updownarrow, \forall)$, $(r, r', \leftrightarrow, \forall)$, $(r, r', \updownarrow, \exists)$ and $(r, r', \leftrightarrow, \exists)$. However, when doing so, we mean the relevant weighted constraint as defined in equations (8), (9), (10) and (11), respectively.

Given an authorization relation $A \subseteq U \times R$, we say a weighted constraint w_c is *user-independent* if, for every permutation σ of U, $w_c(A) = w_c(\sigma(A))$. We have already observed that the APEP constraints in Section 2.1 are user-independent. It is easy to see that the weighted constraints defined above for Valued APEP are also user-independent.

In the remaining sections of this paper, we consider the fixed-parameter tractability of Valued APEP. We will write, for example, APEP$\langle \mathrm{BoD_E} \rangle$ to denote the set of instances of APEP in which the set of constraints C contains only $\mathrm{BoD_E}$ constraints.

3.3 Valued APEP and Valued WSP

We have already observed that WSP is a special case of APEP for certain choices of APEP constraints. Bergé *et al.* also proved that the complexity of some sub-classes of APEP can be reduced to WSP [2, Section 5].

The inputs to Valued WSP include a weighted authorization policy and weighted constraints, and the solution is a plan of minimum weight [12]. Similar arguments to those presented in Section 2.2 can be used to show that Valued WSP is a special case of Valued APEP. In this paper, we will show that some sub-classes of Valued APEP can be reduced to Valued WSP, thereby establishing, via the following result [12, Theorem 1], that those sub-classes of Valued APEP are FPT.

Theorem 3.2. Valued WSP*, when all weighted constraints are user-independent, can be solved in time* $O^*(2^{k \log k})$*, where* $k = |S|$*.*

Following Bertolissi et al. [4] let us consider the following real-world instance of WSP. The goal of the Trip Request Workflow (TRW) is to deal with trips for employees in an organization. TRW has three users u_1, u_2, u_3 and five tasks (steps): Trip request (t_1), Car rental (t_2), Hotel booking (t_3), Flight reservation (t_4), and Trip validation (t_5). Let $T = \{t_i : i = 1, 2, 3, 4, 5\}$. We will impose the following simplifications not affecting our consideration of TRW: we will assume that all tasks are necessary and we will ignore the order in which tasks are executed. TRW has five SoD constraints: $(t_1, t_2, \neq)$, $(t_1, t_4, \neq)$, $(t_2, t_3, \neq)$, $(t_2, t_5, \neq)$, $(t_3, t_5, \neq)$, where $(t_i, t_j, \neq)$ means that t_i and t_j should be performed by different users. Let the cost (weight) of violation of any such constraint be 1. Following Bertolissi et al. [4], we will use a modified authorization policy, but our modification is different: we assume that user u_3 is unavailable. Then the authorization policy is as follows: $\hat{A} = \{(u_1, t_i), (u_2, t_i) : i = 1, 2, 3, 5\} \cup \{(u_1, t_4)\}$. Let the authorization weights be as follows: $\omega(u_1, T') = 0$ for each $T' \subseteq T$, $\omega(u_2, T'') = 0$ for each $T'' \subseteq T \setminus \{t_4\}$ and $\omega(u_2, T''') = 1$ for each $T''' \subseteq T$ such that $t_4 \in T'''$. Observe that this instance of Valued WSP is unsatisfiable as at least one of the three SoD constraints involving t_2, t_3, t_5 cannot be satisfied. Let us assign u_1 to t_2, t_3, t_4 and u_2 to t_1, t_5. Observe that the authorization policy is satisfied and only one constraint is violated. Thus, the weight of the above plan is 1 and so the plan is optimal.

4 t-BOUNDED CONSTRAINTS

In this section we consider instances of Valued APEP having an optimal solution A^* that is small; i.e., $|A^*| \ll |U \times R|$. We start by defining a natural restriction on weighted constraints that implies the existence of a small optimal solution for instances containing only constraints satisfying the restriction. Moreover, checking whether a constraint satisfies the restriction is often easier than checking for the existence of a small optimal solution. This restriction roughly says that if the size of an authorization relation is larger than t, then there are authorizations that are redundant, in the sense that removing those authorizations does not increase the cost of the authorization relation.

Definition 4.1 (t-wbounded)**.** *A set of weighted constraints C is t-wbounded if and only if for each complete authorization relation A such that $|A| > t$ there exists a complete authorization relation A' such that $A' \subseteq A$, $|A'| < |A|$, and $w_C(A') \leq w_C(A)$. We say that a weighted constraint w_c is t-wbounded if the set $\{w_c\}$ is t-wbounded.*

We remark that Bergé *et al.* [2] introduced the notion of $f(k,n)$-bounded user-independent constraints for APEP. While they introduced the notion only for the user-independent constraints it can be easily generalized for any APEP constraint as follows. For an authorization relation A and a user u let us denote by $A - u$ the authorization relation obtained from A by removing all the pairs that include the user u (i.e., the relation $A \setminus \{(u,r) \mid r \in R\}$).

Definition 4.2. *Given a set of resources R and a set of users U, a constraint c is $f(k,n)$-bounded if for each complete authorization relation A which satisfies c, there exists a set U' of size at most $f(k,n)$ such that for each user $u \in (U \setminus U')$, the authorization relation $A - u$ is complete and satisfies c.*

One way to generalize $f(k,n)$-bounded constraints to Valued APEP would be to say that a weighted constraint w_c is $f(k,n)$-bounded if for each complete authorization relations A there exists a set U' of at most $f(k,n)$ users such that for every user $u \in U \setminus U'$, the relation $A - u$ is complete and $w_c(A') \leq w_c(A)$. Given this we can show that our definition covers all constraints covered by Bergé et al.

Lemma 4.3. *If a weighted constraints w_c is $f(k,n)$-bounded, then w_c is $f(k,n) \cdot k$-wbounded. Moreover, if every $c \in C$ is user-independent and $f(k,n)$-bounded then C is $f(k,n) \cdot 2^k \cdot k$-wbounded.*

Proof. Let us consider a complete relation A. If $|A| > f(k,n) \cdot k$, then there are at least $f(k,n) + 1$ users authorized by A. It follows that there exists a user u such that $A(u) \neq \emptyset$ and the authorization relation $A' = A - u$ is complete and $w_c(A') \leq w_c(A)$. But $A' \subseteq A$ and $|A'| < |A|$. Hence w_c is $(f(k,n) \cdot k)$-wbounded. Now, if $|A| > f(k,n) \cdot 2^k \cdot k$, then for some $T \subseteq R$, $T \neq \emptyset$ there are at least $f(k,n) + 1$ users u such that $A(u) = T$. Since every $c \in C$ is user-independent and $f(k,n)$-bounded, it is not difficult to see that for a user u with $A(u) = T$ the authorization relation $A' = A - u$ is complete and $w_c(A') \leq w_c(A)$ for all $c \in C$. Therefore C is $f(k,n) \cdot 2^k \cdot k$-wbounded. □

We can now show that if the set of all constraints in an input instance is t-wbounded, then the size of some optimal solution is indeed bounded by t.

Lemma 4.4. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* Valued APEP *such that C is t-wbounded. Then there exists an optimal solution A^* of $\mathcal{I}$ such that $|A^*| \leq t$.*

Proof. Let A be an optimal solution of $\mathcal{I}$ that minimizes $|A|$. If $|A| \leq t$, then the result follows immediately. For the sake of contradiction, let us assume that $|A| > t$. Since A is a solution, it is complete. Hence by the definition of t-wboundedness, it follows that there exists a complete authorization relation $A' \subseteq A$ such that $A' \subseteq A$, $|A'| < |A|$, and $w_C(A') \leq w_C(A)$. Since A' is a complete authorization relation, it follows that A' is also a solution. Because $A' \subseteq A$, it follows that for all $u \in U$ we have $A'(u) \subseteq A(u)$ and by the monotonicity condition on ω and the definition of the function Ω, we have that $\Omega(A') \leq \Omega(A)$. Finally it follows that $w(A') \leq w(A)$ and A' is also an optimal solution. This however contradicts the choice of the optimal solution A to be an optimal solution that minimizes $|A|$. □

Recall that in WSP the solution is a plan that assigns each step to exactly one user. Hence, we can easily translate an instance of WSP into an APEP instance such that each constraint can be satisfied only if each resource is authorized for exactly one user. Let us call such constraints WSP constraints. It follows that if a relation $A \subseteq U \times R$ satisfies a WSP constraint c, then $|A| = k$ and there are at most k users authorized by A. It follows that in an instance of APEP obtained by a straightforward reduction from WSP we have that every constraint is k-bounded and the set of all constraints is k-wbounded. Therefore, the W[1]-hardness result for WSP established by Wang and Li [30] immediately translates to W[1]-hardness of APEP (and hence also Valued APEP) parameterized by the number of resources k even when the set of all constraints is k-wbounded.

Theorem 4.5. *APEP is W[1]-hard even when restricted to the instances such that C is k-wbounded and every constraint of C is k-bounded.*

Given the above hardness result, from now on we will consider only user-independent constraints. We first show that the user-independent constraints defined in Section 3.2 are t-wbounded. The following lemma significantly improves the existing bounds for Valued APEP$\langle \mathsf{BoD_U}, \mathsf{BoD_E}, \mathsf{SoD_E}, \mathsf{SoD_U}, \mathsf{Card_{UB}} \rangle$ proved in [9].

Lemma 4.6. *Let $\mathcal{I} = (R, U, C, \omega)$ be an input instance to* Valued APEP$\langle \mathsf{BoD_U}, \mathsf{BoD_E}, \mathsf{SoD_E}, \mathsf{SoD_U}, \mathsf{Card_{UB}}, \mathsf{Card_{LB}} \rangle$*. Then, C is $3\tau k\binom{k}{2}$-wbounded, where $\tau = \max_{(r,\geq,t)\in C} t$. Moreover, for any complete authorization relation A, there exists a complete authorization relation $A^* \subseteq A$ such that A^* has at most $3\tau\binom{k}{2}$ users and $w_C(A^*) \leq w_C(A)$.*

Proof. Let A be a complete authorization relation. If A has at most $3\tau\binom{k}{2}$ users, then $|A| \leq 3\tau k\binom{k}{2}$ and we are done. Suppose that A has more than $3\tau\binom{k}{2}$ users. We define an equivalence relation $\equiv$ in R as follows: $r \equiv r'$ if and only if $A(r) = A(r')$. This equivalence relation yields a partition of R, $R_1 \uplus \ldots \uplus R_p$. Note that for any $r \in R_i$, we have that $A(R_i) = A(r)$. For any $i, j \in [p]$ with $i \neq j$, we will consider $A(R_i) \setminus A(R_j)$, $A(R_j) \setminus A(R_i)$, and $A(R_i) \cap A(R_j)$. Since $A(R_i) \neq A(R_j)$, either $A(R_i) \setminus A(R_j) \neq \emptyset$ or $A(R_j) \setminus A(R_i) \neq \emptyset$ or both. We mark some users from A as follows to construct a new authorization relation A^*. If $A(R_i) \setminus A(R_j) \neq \emptyset$, we mark $\min(\tau, |A(R_i) \setminus A(R_j)|)$ many users in $A(R_i) \setminus A(R_j)$; in particular if $|A(R_i) \setminus A(R_j)| \leq \tau$, then we mark all the users in $A(R_i) \setminus A(R_j)$. Similarly, for non-empty $A(R_j) \setminus A(R_i)$ and $A(R_i) \cap A(R_j)$. We repeat this process for all pairs $i, j \in [p]$ with $i \neq j$. We delete all unmarked users from A and output A^* as the new authorization relation. Clearly, there are at most $3\tau\binom{k}{2}$ marked users in A^*. Thus, $|A^*| \leq 3\tau k\binom{k}{2}$. Observe that for any marked user u, $A^*(u) = A(u)$ and for any $i \in [p]$, for any $r, r' \in R_i$, $A^*(r) = A^*(r')$.

We first argue that for two distinct $i, j \in [p]$, $A^*(R_i) \cap A^*(R_j) \neq \emptyset$ if and only if $A(R_i) \cap A(R_j) \neq \emptyset$. As $A^* \subseteq A$, if there exists $u \in A^*(R_i) \cap A^*(R_j)$, then the same user $u \in A(R_i) \cap A(R_j)$. On the other hand, let $A(R_i) \cap A(R_j) \neq \emptyset$. Then, there exists $u \in A(R_j) \cap A(R_j)$ that we have marked using our marking scheme. So, $A^*(R_i) \cap A^*(R_j) \neq \emptyset$.

Next we argue that for two distinct $i, j \in [p]$, $A^*(R_i) \neq A^*(R_j)$. By the definition of $R = R_1 \uplus \ldots \uplus R_p$, we have $A(R_i) \neq A(R_j)$. Therefore, either there exists $u \in A(R_i) \setminus A(R_j)$ or there exists $u' \in A(R_j) \setminus A(R_i)$ or both. If there exists $u \in A(R_i) \setminus A(R_j)$, then we have marked at least one such u. If there exists $u \in A(R_j) \setminus A(R_i)$, then we have marked at least one such u. As for any marked user u, $A^*(u) = A(u)$, $A^*(R_j) \neq A^*(R_i)$.

By the arguments above, we have the following:

- $A^* \subseteq A$,
- $A(r) = A(r')$ if and only if $A^*(r) = A^*(r')$, and
- $A(r) \cap A(r') \neq \emptyset$ if and only if $A^*(r) \cap A^*(r') \neq \emptyset$.

Consider a constraint $c = (r, r', \leftrightarrow, \forall) \in C$. By (9) in Section 3, $w_c(A) = f_c(\mathrm{maxdiff(A, r, r')})$, where $\mathrm{maxdiff(A, r, r')} = \max\{|\mathrm{A(r)} \setminus \mathrm{A(r')}|, |\mathrm{A(r')} \setminus \mathrm{A(r)}|\}$. Observe that $\mathrm{maxdiff(A^*, r, r')} \leq \mathrm{maxdiff(A, r, r')}$. Hence, $w_c(A^*) \leq w_c(A)$.

Consider a constraint $c = (r, r', \leftrightarrow, \exists) \in C$. By (11) in Section 3, if $A(r) \cap A(r') \neq \emptyset$ then $w_c(A) = 0$; otherwise, $w_c(A) = \ell_c > 0$. As we have proved, $A^*(r) \cap A^*(r') \neq \emptyset$ if and only if $A(r) \cap A(r') \neq \emptyset$. Hence, $w_c(A) = w_c(A^*) = \ell_c$.

Consider a constraint $c = (r, r', \updownarrow, \forall) \in C$. By (8) in Section 3, $w_c(A) = f_c(|A(r) \cap A(r')|)$. We have shown that $A^*(r) \cap A^*(r') = \emptyset$ if and only if $A(r) \cap A(r') = \emptyset$. Moreover, $|A^*(r) \cap A^*(r')| \leq |A(r) \cap A(r')|$. Hence, $w_c(A^*) \leq w_c(A)$.

Consider a constraint $c = (r, r', \updownarrow, \exists) \in C$. By (10) in Section 3, if $A(r) \neq A(r')$ then $w_c(A) = 0$; otherwise $w_c(A) = \ell_c > 0$. We have proved that $A(r) \neq A(r')$ if and only if $A^*(r) \neq A^*(r')$. It implies that $w_c(A^*) = w_c(A) = \ell_c$.

Consider a constraint $c = (r, \leq, t)$. By (6) in Section 3, $w_c(A) = f_c(|A(r)| - t)$. Observe that $A^*(r) \subseteq A(r)$. Hence, $w_c(A^*) \leq w_c(A)$.

Finally, consider a constraint $c = (r, \geq, t) \in C$. By (7) in Section 3, $w_c(A) = f_c(t - |A(r)|)$. Note that $t \leq \tau$. Let $i, j \in [p]$ be such that $r \in R_i$ and $j \neq i$. Notice that $A(r) = (A(R_i) \setminus A(R_j)) \cup (A(R_i) \cap A(R_j))$ and we marked $\min(\tau, |A(R_i) \setminus A(R_j)|)$ users in $A(R_i) \setminus A(R_j)$ and $\min(\tau, |A(R_i) \cap A(R_j)|)$ users in $A(R_i) \cap A(R_j)$. Therefore, if $|A(r)| \leq \tau$, then $A^*(r) = A(r)$ and $w_c(A^*) = w_c(A)$. Otherwise $|A(r)| \geq \tau$ implying $|A^*(r)| \geq \tau \geq t$ and $w_c(A^*) = w_c(A) = 0$.

Thus, we conclude that $w_C(A^*) \leq w_C(A)$. ◻

By definition, if we have user-independent constraints, then we do not need to know which particular users are assigned to resources in order to determine the constraint weight of some authorization relation A. Instead, it suffices to know for each set $T \subset R$ how many users u are authorized by A precisely for the set T, i.e., the size of the set $\{u \in U \mid A(u) = T\}$. This leads us to the following definition of the *user profile of an authorization relation.* Lemma 4.8 confirms the intuition behind the definition: if we have a user-independent constraint, then two authorization relations with the same user profile yield the same constraint weight.

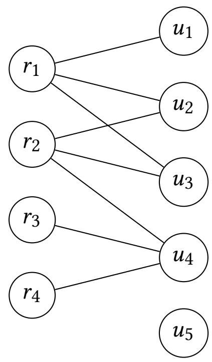
Definition 4.7 (user profile). *For a set of resources R, a set of users U, and an authorization relation $A \subseteq U \times R$, the* user profile *of the authorization relation A is the function* $\mathrm{usr}_A : 2^R \to \mathbb{N}$, *where* $\mathrm{usr}_A(T)$ *is defined to be* $|\{u \in U \mid A(u) = T\}|$.

Note that $\mathrm{usr}_A(T)$ is not the same as $|A(T)|$. The integer $\mathrm{usr}_A(T)$ is the number of all users that are authorized for all resources in T and nothing else, while $A(T)$ is the set of users that are authorized for at least one resource in T.

An example of a user profile is given in Figure 1b. Here $R = \{r_1, r_2, r_3, r_4\}$ and $U = \{u_1, u_2, u_3, u_4, u_5\}$. Let A be an authorization relation, shown in Figure 1a in the form of a bipartite graph, such that $A(r_1) = \{u_1, u_2, u_3\}$, $A(r_2) = \{u_2, u_3, u_4\}$, $A(r_3) = \{u_4\}$ and $A(r_4) = \{u_4\}$. Then Figure 1b shows the user profile of A.

Lemma 4.8. *Let U be a set of users, R a set of resources, (c, w_c) a user-independent weighted constraint and $A_1, A_2 \subseteq U \times R$ two authorization relations such that $\mathrm{usr}_{A_1}(T) = \mathrm{usr}_{A_2}(T)$ for all $T \subseteq R$. Then $w_c(A_1) = w_c(A_2)$.*

Proof. We will define a permutation $\sigma : U \to U$ such that $\sigma(A_1) = A_2$. The lemma then immediately follows from the definition of user-independence. For $i \in \{1, 2\}$ and $T \subseteq R$, let U_T^i be the set of users that are assigned by A_i precisely to the resources in T and nothing else. That is $U_T^i = \{u \in U \mid A_i(u) = T\}$. Now, let us fix for each U_T^i an arbitrary ordering of the users in U_T^i and let $u_{T,j}^i$ for $j \in [|U_T^i|]$ denote the j-th user in U_T^i. Note that for all $T \subseteq R$, we have $\mathrm{usr}_{A_1}(T) = \mathrm{usr}_{A_2}(T)$ by the assumptions of the lemma and hence $|U_T^1| = |U_T^2|$. Moreover, each user in U is assigned exactly one (possibly empty) subset of resources in each of the authorization relations A_1 and A_2. Hence the sets $\bigcup_{T \subseteq R}\{U_T^1\}$ and $\bigcup_{T \subseteq R}\{U_T^2\}$ are both partitions of U. We are now ready to define the permutation σ as $\sigma(u_{T,j}^1) = u_{T,j}^2$ for all $T \subseteq R$, $j \in [|U_T^1|]$. It remains to show that $\sigma(A_1) = A_2$. By the definition of the users $u_{T,j}^1$ and $u_{T,j}^2$, we get that for all $T \subseteq R$, all $j \in [|U_T^1|]$, and all $r \in R$ we have $(u_{T,j}^1, r) \in A_1$ if and only if $r \in T$ if and only if $(u_{T,j}^2, r) \in A_2$ and the lemma follows. ◻



(a) Authorization relation

T	$\mathrm{usr}_A(T)$
$\{r_1\}$	1
$\{r_1, r_2\}$	2
$\{r_2, r_3, r_4\}$	1
$\emptyset$	1
Other $T \subseteq R$	0

(b) User profile

Fig. 1. An example of an authorization relation and the corresponding user profile.

Lemma 4.9. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* VALUED APEP *such that all constraints in C are user-independent and let* $\mathrm{usr} : 2^R \to \mathbb{N}$ *be a user profile. Then there exists an algorithm that finds a relation A which minimizes $w(A)$ among all relations with user profile* usr.

PROOF. It follows from Lemma 4.8 and the fact that all constraints in C are user-independent that $w_C(A)$ only depends on the user profile of A and hence we only need to find an authorization relation A with user profile $\mathrm{usr}_A = \mathrm{usr}$ such that $\Omega(A)$ is minimized.

Note that if $\sum_{T \subseteq R} \mathrm{usr}(T) \neq |U|$, then there is no authorization relation with given user profile. This is because for every authorization relation A and every user u, the set $A(u)$ is defined as is a (possibly empty) subset of R. Hence from now on we assume that $\sum_{T \subseteq R} \mathrm{usr}(T) = |U|$. We start by creating a weighted complete bipartite graph $G = (V_1 \cup V_2, E)$, with parts V_1, V_2 such that $V_1 = U$ and V_2 contains for each $T \subseteq R$ a set of $\mathrm{usr}(T)$ vertices; let us denote these vertices $\{v_1^T, v_2^T, \ldots, v_{\mathrm{usr}(T)}^T\}$. For a user $u \in U$ and a vertex v_i^T, the weight of the edge uv_i^T is defined as $w(uv_i^T) = \omega(u, T)$. Since $\sum_{T \subseteq R} \mathrm{usr}(T) = |U|$, it follows that $|V_1| = |V_2|$. We show that there is a correspondence between perfect matchings of the graph G and authorization relations with user profile usr.

First, let A be an authorization relation such that $\mathrm{usr}_A = \mathrm{usr}$. Then, we can get a perfect matching M_A of G of weight $\Omega(A)$ as follows. Because, $\mathrm{usr}_A = \mathrm{usr}$, we have that for every $T \subseteq R$ there are exactly $\mathrm{usr}(T)$ many users $u \in U$ such that $A(u) = T$. Hence for every $T \subseteq R$ there is a perfect matching M_A^T between these users and vertices $\{v_1^T, v_2^T, \ldots, v_{\mathrm{usr}(T)}^T\}$. Moreover, the cost of an edge between a user $u \in U$ such that $A(u) = T$ and a vertex v_j^T, $j \in \mathrm{usr}(T)$, is $\omega(u, T)$ which is precisely the contribution of the user u to $\Omega(A)$. Hence the cost of the matching $M_A = \bigcup_{T \subseteq R} M_A^T$ is precisely $\Omega(A)$.

On the other hand if M is a perfect matching in G, then we can define an authorization relation A_M as $(u, r) \in A_M$, if and only if u is matched to a vertex v_i^T with $r \in T$. Clearly, every user u is then matched by M to a vertex v_i^T such that $A_M(u) = T$ and weight of the edge in M incident to u is precisely $\omega(u, A_M(u))$, which is the contribution of u to $\Omega(A)$.

It follows that G has a perfect matching of cost W if and only if there is an authorization relation A with user profile usr and $\Omega(A) = W$ and given a perfect matching of G, we can easily find such an authorization relation. Therefore, to finish the proof of the lemma we only need to compute a minimum cost perfect matching in the weighted bipartite graph G, which can be done using the well-known Hungarian method in $O(mn)$ time [23], where n is the number of vertices and m is the number of edges in G. □

The next ingredient required to prove our main result (Theorem 4.14) is the fact that the number of all possible user profiles for all authorization relations of size at most t is small and can be efficiently enumerated. Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of VALUED APEP and $A \subseteq U \times R$ an authorization relation. Then for the user profile usr_A of A we have that $\sum_{T\subseteq R} |T| \cdot \mathrm{usr}_A(T) = |A|$ and $\sum_{T\subseteq R} \mathrm{usr}_A(T) = |U|$. Moreover, if A is complete, then $t \geq k$. Note that the number of users in an optimal solution for a t-wbounded set of weighted constraints is at most t by Lemma 4.4. However, sometimes we are able to show that the number of users in an optimal solution is actually significantly smaller than the bound t such that the set of weighted constraints C is t-wbounded (see, e.g., Lemma 4.6). Moreover, if usr_A is a user profile of an authorization relation with at most ℓ users, then $\sum_{T\subseteq R, T\neq\emptyset} \mathrm{usr}_A(T) \leq \ell$. The following lemma will be useful because we are only interested in complete authorization relations of size at most t that use at most $\ell \leq t$ users.

Lemma 4.10. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* VALUED APEP *such that $|R| = k$ and let $\ell \in \mathbb{N}$. Then the number of possible user profiles, $\mathrm{usr} : 2^R \to \mathbb{N}$, such that $\sum_{T\subseteq R, T\neq\emptyset} \mathrm{usr}(T) \leq \ell$ is $\binom{\ell+2^k-1}{\ell}$. Moreover, we can enumerate all such functions in time $O^*(\binom{\ell+2^k-1}{\ell})$.*

PROOF. It is well known that the number of weak compositions of a natural number q into p parts (the number of ways we can assign non-negative integers to the variables $x_1, x_2, \ldots, x_p$ such that $\sum_{i=1}^{p} x_i = q$) is precisely $\binom{p+q-1}{q-1} = \binom{p+q-1}{p-1}$ (see, e.g., [20]). Note that because $\sum_{T\subseteq R} \mathrm{usr}(T) = |U|$, each user profile usr is determined by assigning $\mathrm{usr}(T)$ for all $T \neq \emptyset$. It is not difficult to see that the number of ways in which we can assign $\mathrm{usr}(T)$ for all $T \neq \emptyset$ such that $\sum_{T\subseteq R, T\neq\emptyset} \mathrm{usr}(T) \leq \ell$ is the same as the number of weak partitions of ℓ into 2^k parts - each of the first $2^k - 1$ parts is identified with one of $2^k - 1$ sets $T \subseteq R$ such that $T \neq \emptyset$. The last part is then a "slack" part that allows $\sum_{T\subseteq R, T\neq\emptyset} \mathrm{usr}(T)$ to be also smaller than ℓ. It follows that the number of possible user profiles is at most $\binom{\ell+2^k-1}{\ell}$. To enumerate them in $O^*(\binom{\ell+2^k-1}{\ell})$ time we can do the following branching algorithm: We fix some order $T_1, T_2, \ldots, T_{2^k-1}$ of the non-empty subsets of R. We first branch on $\ell + 1$ possibilities for $\mathrm{usr}(T_1)$, then we branch on $\ell + 1 - \mathrm{usr}(T_1)$ possibilities for $\mathrm{usr}(T_2)$, and so on, until we branch on $\ell + 1 - \sum_{i\in[2^k-2]} \mathrm{usr}(T_i)$ possibilities for $\mathrm{usr}(T_{2^k-1})$. Afterwards, we compute $\mathrm{usr}(\emptyset)$ from $\sum_{T\subseteq R} \mathrm{usr}(T) = |U|$. Each leaf of the branching tree gives us a different possible user profile and we spend polynomial time in each branch. Hence the running time of the enumeration algorithm is $O^*(\binom{\ell+2^k-1}{\ell})$. □

Because the number of possible user profiles that authorize at most ℓ users appears in the running time of our algorithms, it will be useful to keep in mind the following two simple observations about the combinatorial number $\binom{\ell+2^k-1}{\ell}$.

Observation 4.11. $\binom{\ell+2^k-1}{\ell} \leq 2^{\ell+2^k-1} \leq 4^{\max(\ell, 2^k-1)}$.

Observation 4.12. *If $\ell \geq 4$, then $\binom{\ell+2^k-1}{\ell} \leq \min(2^{\ell k}, \ell^{2^k-1}) + 1$.*

PROOF. If $k = 0$, then $\binom{\ell+2^k-1}{\ell} = \binom{\ell}{\ell} = 1 \leq \min(2^{\ell\cdot 0}, \ell^{2^0-1}) + 1$. If $k = 1$, then $\binom{\ell+2^k-1}{\ell} = \ell+1 \leq \min(2^{\ell\cdot 1}, \ell^{2^1-1}) + 1$. If $k = 2$, then $\binom{\ell+2^k-1}{\ell} = \binom{\ell+3}{3} = \frac{\ell^3+6\ell^2+11\ell+1}{6}$ and since $\ell \geq 4$, it follows that $\binom{\ell+3}{3} \leq \ell^3 \leq 2^{2\ell}$. From now on, let us assume that $k \geq 3$ and $\ell \geq 4$. We distinguish between two cases depending on whether $\ell < 2^k$ or $\ell \geq 2^k$. Let us first consider $\ell \geq 2^k$. Note that in this case $\ell^{2^k-1} \leq 2^{\ell k}$ and because $k \geq 2$ we have $\binom{\ell+2^k-1}{\ell} = \binom{\ell+2^k-1}{2^k-1} \leq \frac{(\ell+2^k-1)^{2^k-1}}{(2^k-1)!} \leq \frac{2^{2^k-1}}{(2^k-1)!} \cdot \ell^{2^k-1} \leq \ell^{2^k-1}$. On the other hand, let us now assume $\ell \leq 2^k - 1$. Note that, because $k \geq 2$ and $\ell \geq 3$, it holds that $(2^k-1)^\ell \leq \ell^{2^k-1}$. Furthermore, $(2^k-1)^\ell < 2^{\ell k}$. Then $\binom{\ell+2^k-1}{\ell} \leq \frac{(\ell+2^k-1)^\ell}{\ell!} \leq \frac{2^\ell}{\ell!} \cdot (2^k-1)^\ell \leq (2^k-1)^\ell$. □

We are now ready to state and prove the main lemma of this section, which establishes that there exists an FPT algorithm that finds the best solution among all solutions that authorize at most ℓ users. In particular, this algorithm finds an optimal solution for the case of user-independent t-wbounded constraints.

Lemma 4.13. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* Valued APEP *such that all weighted constraints in C are user-independent and let $\ell \in \mathbb{N}$. Then there exists an algorithm that in time $O^*(\binom{\ell+2^k-1}{\ell})$ computes a complete authorization relation A such that $w(A) \leq w(A')$ for every complete authorization relation $A' \subseteq U \times R$ that authorizes at most ℓ users for some resource in R.*

Proof. Note that it suffices to compute such an authorization relation A that also authorizes at most ℓ users. Let A^* be one such complete authorization relation that satisfies the statement of the theorem. Let $\mathrm{usr}_{A^*} : 2^R \to \mathbb{N}$ be the user profile of A^*. Observe that $\sum_{T \subseteq R, T \neq \emptyset} \mathrm{usr}_{A^*}(T) \leq \ell$. Moreover, observe that $\sum_{T \subseteq R} \mathrm{usr}_{A^*}(T) = |U|$, as every user in U is assigned to precisely one subset of resources by A^*. Furthermore, notice that since A^* is complete, for all $r \in R$ we have that $\sum_{\{r\} \subseteq T \subseteq R} \mathrm{usr}_{A^*}(T) \geq 1$ and we may restrict our attention to user profiles that also satisfy $\sum_{\{r\} \subseteq T \subseteq R} \mathrm{usr}_{A^*}(T) \geq 1$ for all $r \in R$. By Lemma 4.10, there exist $\binom{\ell+2^k-1}{\ell}$ different functions (possible user profiles) $\mathrm{usr} : 2^R \to \mathbb{N}$ such that $\sum_{T \subseteq R, T \neq \emptyset} \mathrm{usr}(T) \leq \ell$ and $\sum_{T \subseteq R} \mathrm{usr}(T) = |U|$. Moreover, we can enumerate all of them in time $O^*(\binom{\ell+2^k-1}{\ell})$.

Now, let $\mathcal{P}$ be the set of all such possible user profiles obtained by Lemma 4.10 that also satisfy $\sum_{\{r\} \subseteq T \subseteq R} \mathrm{usr}_{A^*}(T) \geq 1$ for all $r \in R$. Since $\mathcal{P}$ is a subset of functions computed by Lemma 4.10, it follows that $|\mathcal{P}| \leq \binom{\ell+2^k-1}{\ell}$. Moreover, it is easy to see that $\mathrm{usr}_{A^*} \in \mathcal{P}$. The algorithm then branches on all possible profiles in $\mathcal{P}$ and for a profile $\mathrm{usr}_i \in \mathcal{P}$, $i \in [|\mathcal{P}|]$, it computes an authorization relation A_i such that $\mathrm{usr}_{A_i} = \mathrm{usr}_i$ and $w(A_i)$ is minimized, which can be done in polynomial time by Lemma 4.9. Finally, the algorithm outputs the authorization relation A_i for the user profile usr_i that minimizes $w(A_i)$ among all $\mathrm{usr}_i \in \mathcal{P}$. The running time of the whole algorithm is $O^*(\binom{\ell+2^k-1}{\ell})$.

To establish correctness, first notice that for all $i \in [|\mathcal{P}|]$ we have $\sum_{\{r\} \subseteq T \subseteq R} \mathrm{usr}_i(T) \geq 1$ for all $r \in R$, so the authorization relation A_i is complete. Furthermore, recall that $\mathrm{usr}_{A^*} \in \mathcal{P}$. For $i \in [|\mathcal{P}|]$ such that $\mathrm{usr}_i = \mathrm{usr}_{A^*}$, we have that $w(A_i) \leq w(A^*) \leq w(A')$ for all complete authorization relations $A' \subseteq U \times R$ that authorizes at most ℓ users for some resource in R. ◻

Note that it follows from Lemma 4.4 that given an instance $\mathcal{I} = (R, U, C, \omega)$ of Valued APEP such that all weighted constraints in C are user-independent and C is t-wbounded there exists an optimal solution A^* of $\mathcal{I}$ such that $|A^*| \leq t$. Moreover, $|A^*| \leq t$ implies that A^* authorizes at most t users for some resource. Hence, in combination with Observation 4.12, we immediately obtain the main result of this section as a corollary, which establishes that there exists an FPT algorithm for the case of user-independent t-wbounded constraints.

Theorem 4.14. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* Valued APEP *such that all weighted constraints in C are user-independent and C is t-wbounded. Then there exists an algorithm solving $\mathcal{I}$ in time $O^*(\binom{t+2^k-1}{t}) = O^*(\min(2^{tk}, t^{2^k-1})) = O^*(2^{\min(kt, (2^k-1)\log t)})$.*

Using Theorem 4.14 and Lemma 4.6, we have the following:

Corollary 4.15. *Let $\tau = \max_{(r,\geq,t)\in C} t$. Then* Valued APEP$\langle \mathrm{BoD_U}, \mathrm{BoD_E}, \mathrm{SoD_E}, \mathrm{SoD_U}, \mathrm{Card_{UB}}, \mathrm{Card_{LB}} \rangle$ *can be solved in $O^*(8^{\tau k^2 \binom{k}{2}})$ time. Thus,* Valued APEP$\langle \mathrm{BoD_U}, \mathrm{BoD_E}, \mathrm{SoD_E}, \mathrm{SoD_U}, \mathrm{Card_{UB}}, \mathrm{Card_{LB}} \rangle$ *parameterized by $k + \tau$ is FPT.*

Proof. As $t \leq \tau 3k\binom{k}{2}$, we have $2^{kt} \leq 8^{k^2\binom{k}{2}}$ completing the proof. ◻

In Sections 5 and 6 we develop more specialized algorithms that solve Valued APEP more efficiently for instances where all constraints are from some specific subset of the above constraints. We conclude this section by showing that restricting attention to user-independent constraints is not sufficient to obtain an FPT algorithm parameterized by the number of resources k even for APEP. Because all weighted constraints are necessarily $(k \cdot |U|)$-wbounded, as a corollary of our conditional lower-bound, we will also show that, unless the Exponential Time Hypothesis (ETH) fails, the algorithm in Theorem 4.14 is, in a sense, the best that we can hope for from an algorithm that can solve Valued APEP with arbitrary user-independent t-wbounded weighted constraints.

Because all the constraints we saw so far were 2^k-wbounded, we will need to introduce new, more restrictive, constraints to obtain our W[2]-hardness and ETH lower-bound. As we consider only user-independent constraints, it is natural for a constraint c to be a function of the user profile of A. To this end, we define a new type of user-independent constraint $c = (\tau, X, \bigvee)$, where $\tau \subseteq 2^R$ and $X \subseteq \mathbb{N}$. The constraint $(\tau, X, \bigvee)$ is satisfied if and only if there exist $T \in \tau$ and $x \in X$ such that $\mathrm{usr}_A(T) = x$. Less formally, c is satisfied if and only if some specified number of users (in X) is authorized for some specified set of resources (in τ). To obtain our ETH lower-bound we make use of the following well-known result in parameterized complexity:

Theorem 4.16 ([25]). *Assuming ETH, there is no $f(k)n^{o(k)}$-time algorithm for Dominating Set, where n is the number of the vertices of the input graph, k is the size of the output set, and f is an arbitrary computable function.*

Given the above theorem, we are ready to prove the main negative result of this section.

Theorem 4.17. *APEP is W[2]-hard and, assuming ETH, there is no $f(|R|) \cdot |\mathcal{I}|^{o(2^{|R|})}$-time algorithm solving APEP even when all constraints are user-independent and the base authorization relation is $U \times R$.*

Henceforth, we will write $[k]$ to denote $\{1, \ldots, k\}$. For a graph $G = (E, V)$ and vertex $x \in V$, $N_G(x) = \{y \in V \mid xy \in E\}$ is the set of vertices adjacent to x in G; for a set $S \subseteq V(G)$, $N(S) = \bigcup_{x \in S} N_G(x) \setminus S$.

Proof. To prove the theorem we give a reduction from the Dominating Set problem. Let (G, k) be an instance of the Dominating Set problem. Let $|V(G)| = n$ and let $V(G) = \{v_1, v_2, \ldots, v_n\}$; that is we fix some arbitrary ordering of the vertices in G and each vertex of G is uniquely identified by its position in this ordering (index of the vertex). For a vertex $v_i \in V(G)$ we let the set $X_i = \{i\} \cup \{j \mid v_j \in N_G(v_i)\}$. In other words, for a vertex $v_i \in V(G)$, the set X_i is the set of indices of the vertices in the closed neighbourhood of v_i. The aim of the Dominating Set problem is then to decide whether G has a set S of at most k vertices such that for all $i \in [n]$ the set X_i contains an index of some vertex in S.

Let $\mathcal{I} = (U, R, \hat{A}, C)$ be an instance of APEP such that

- $R = \{r_1, \ldots, r_\ell\}$ such that $2^{\ell-1} \leq k < 2^\ell$,
- $|U| = k \cdot n$,
- $C = \bigcup_{i \in [n]} \{(\tau, X_i, \bigvee)\}$, where $\tau \subset 2^R$ such that $\emptyset \notin \tau$ and $|\tau| = k$, and
- $\hat{A} = U \times R$.

We prove that (G, k) is a YES-instance of Dominating Set if and only if $\mathcal{I}$ is a YES-instance of APEP. Let $\tau = \{T_1, T_2, \ldots, T_k\}$. Observe that because $2^{\ell-1} \leq k$ and $T_i \neq T_j$ for $i \neq j$, it follows that every resource appears in T_i for some $i \in [k]$.

Let $S = \{v_{q_1}, v_{q_2}, \ldots, v_{q_k}\}$ be a dominating set of G of size k (note that if we have a dominating set of size at most k, then we have a dominating set of size exactly k). Let A be an authorization relation such that $\mathrm{usr}_A(T_i) = q_i$. Because $|U| = k \cdot n$ and $1 \leq q_i \leq n$ for all $i \in [k]$, it is easy to construct such an authorization relation. For each $i \in [k]$ we simply select q_i many fresh users u such that $A(u) = T_i$ and leave the remaining users not assigned to any resources ($A(u) = \emptyset$). Because $2^{\ell-1} \leq k$ and for all $T_i \in \tau$ we have $\mathrm{usr}_A(T_i) \geq 1$, it is easy to see that A is a complete authorization relation. Since $\hat{A} = U \times R$, A is authorized. It remains to show that A is eligible w.r.t. C. Consider the constraint $c_i = (\tau, X_i, \bigvee)$. Since S is a dominating set, the closed neighbourhood of v_i contains a vertex $v_{q_j} \in S$. But then $q_j \in X_i$, $T_j \in \tau$, and $\mathrm{usr}_A(T_j) = q_j$, hence c_i is satisfied.

On the other hand let A be valid w.r.t. $\hat{A}$. We obtain a dominating set of G of size at most k as follows. Without loss of generality, let as assume that if $i < j$, then $\mathrm{usr}_A(T_i) \geq \mathrm{usr}_A(T_j)$ and let $k' \in [k]$ be such that $\mathrm{usr}_A(T_{k'}) \geq 1$ and $\mathrm{usr}_A(T_{k'+1}) = 0$ (note that if $\mathrm{usr}_A(T_k) \geq 1$, then $k' = k$). We let $S = \bigcup_{i \in [k']} \{v_{\mathrm{usr}_A(T_i)}\}$. We claim that S is a dominating set (clearly $|S| = k' \leq k$). Let v_i be arbitrary vertex in $V(G) \setminus S$. Consider the constraint $c_i = (\tau, X_i, \bigvee)$. Clearly c_i is satisfied and there exists $T_j \in \tau$ and $x \in X_i$ such that $\mathrm{usr}_A(T_j) = x$. But by definition of X_i, $x \geq 1$ and v_x is a neighbour of v_i. Moreover, by the definition of S, we have $v_x = v_{\mathrm{usr}_A(T_j)} \in S$. It follows that S is a dominating set.

Now for each $i \in [n]$, the set X_i has size at most n and τ has size $k \leq n$, so the size of the instance $\mathcal{I}$ is polynomial in n. Moreover $2^{\ell-1} \leq k < 2^{\ell}$, hence APEP is W[2]-hard parameterized by $|R|$ and an $f(|R|) \cdot |\mathcal{I}|^{o(2^{|R|})}$ time algorithm for APEP yields an $f(k)n^{o(k)}$ time algorithm for Dominating Set, and the result follows from Theorem 4.16. ◻

The set C is trivially $(|R| \cdot |U|)$-wbounded for every Valued APEP instance $\mathcal{I} = (R, U, C, \omega)$, so we obtain the following result, which asserts that the lower bound asymptotically matches the running time of the algorithm from Theorem 4.14.

Corollary 4.18. *Assuming ETH, there is no $t^{o(2^{|R|})} \cdot n^{O(1)}$ time algorithm that given an instance $\mathcal{I} = (R, U, C, \omega)$ of* Valued APEP *such that all constraints in C are user-independent and C is t-wbounded computes an optimal solution for $\mathcal{I}$.*

5 $\mathrm{SoD_U}$ AND $\mathrm{BoD_U}$ CONSTRAINTS

In this section, we will consider Valued APEP, where all constraints are only $\mathrm{BoD_U}$ and $\mathrm{SoD_U}$. We will show how to reduce it to Valued WSP with user-independent constraints, with the number of steps equal to the number k of resources in Valued APEP. As a result, we will be able to obtain an algorithm for Valued APEP with only $\mathrm{BoD_U}$ and $\mathrm{SoD_U}$ constraints of running time $O^*(2^{k \log k})$.

Let us start with Valued APEP$\langle \mathrm{SoD_U} \rangle$. Recall that the weighted version of an $\mathrm{SoD_U}$ constraint $(r, r', \updownarrow, \forall)$ is $w_c(A) = f_c(|A(r) \cap A(r')|)$ for some monotonically increasing function f_c.

The weight of a binary SoD constraint $c = (s', s'', \neq)$ in Valued WSP is 0 if and only if steps s' and s'' are assigned to different users. Valued WSP using only SoD constraints of this form will be denoted by Valued WSP($\neq$).

Lemma 5.1. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* Valued APEP$\langle \mathrm{SoD_U} \rangle$ *and let A^* be an optimal solution of $\mathcal{I}$. Let A' be arbitrary authorization relation such that $A' \subseteq A^*$ and $|A'(r)| = 1$ for every $r \in R$. Then A' is an optimal solution of $\mathcal{I}$. Moreover, in polynomial time $\mathcal{I}$ can be reduced to an instance $\mathcal{I}'$ of* Valued WSP($\neq$) *such that the weights of optimal solutions of $\mathcal{I}$ and $\mathcal{I}'$ are equal.*

Proof. Let A' be an arbitrary relation such that $A' \subseteq A^*$ and $|A'(r)| = 1$ for every $r \in R$. By definition, A' is complete. By (2) and (3), $A' \subseteq A^*$ implies $\Omega(A') \leq \Omega(A^*)$. Let $c = (r, r', \updownarrow, \forall) \in C$. Since $A'(r) \cap A'(r') \subseteq A^*(r) \cap A^*(r')$, $w_c(A) = f_c(|A(r) \cap A(r')|)$ and f_c is non-decreasing, we have $w_c(A') \leq w_c(A^*)$. Thus, $\Omega(A') + w_C(A') \leq \Omega(A^*) + w_C(A^*)$. Since A^* is optimal, A' is optimal, too.

Define an instance $\mathcal{I}'$ of Valued WSP($\neq$) as follows: the set of steps is R, the set of users is U, and $(r, r', \neq)$ is a constraint of $\mathcal{I}'$ if $(r, r', \updownarrow, \forall)$ is a constraint of $\mathcal{I}$. The weight of $(r, r', \neq)$ equals $w_c(A') = f_c(1)$ (recall that $|A'(r)| = 1$

for all r) and the weights of (u,T), $u \in U, T \subseteq R$, in both $\mathcal{I}$ and $\mathcal{I}'$ are equal. Observe that $\pi : R \to U$ defined by $\pi(r) = A'(r)$ is an optimal plan of $\mathcal{I}'$. Thus, the optimal solution of $\mathcal{I}$ has the same weight as that of $\mathcal{I}'$. ◻

We next consider Valued APEP$\langle \mathrm{BoD_U}, \mathrm{SoD_U}\rangle$. Recall that the weighted version of a $\mathrm{BoD_U}$ constraint $(r, r', \leftrightarrow, \forall)$ is given by $f_c(\max\{|A(r) \setminus A(r')|, |A(r') \setminus A(r)|\})$ for some monotonically increasing function f_c.

The weight of binary BoD constraint $c = (s', s'', =)$ in Valued WSP is 0 if and only if steps s' and s'' are assigned the same user. Valued WSP(=) denotes Valued WSP containing only BoD constraints; Valued WSP(=, ≠) denotes Valued WSP containing only SoD and BoD constraints. In fact, Valued WSP(=) is already NP-hard which follows from Theorem 6.4 of [8]. This theorem, in particular, shows that Valued WSP(=) is NP-hard even if the weights are restricted as follows: $w_c(\pi) = 1$ if a plan π falsifies a constraint c, $\omega(u, r) = \infty$ if $(u, r) \notin \hat{A}$.

Lemma 5.2. *Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of* Valued APEP$\langle \mathrm{BoD_U}, \mathrm{SoD_U}\rangle$ *and let A^* be an optimal solution of $\mathcal{I}$. There is an optimal solution A' of $\mathcal{I}$ such that $A' \subseteq A^*$ and $|A'(r)| = 1$ for every $r \in R$. Moreover, in polynomial time $\mathcal{I}$ can be reduced to an instance $\mathcal{I}'$ of* Valued WSP(=, ≠).

Proof. Consider an optimal solution A^* of $\mathcal{I}$ and define A' as follows. We first define an equivalence relation $\cong$ on R, where $r \cong r'$ if and only if $A^*(r) = A^*(r')$. This gives a partition $R = R_1 \uplus \ldots \uplus R_p$ such that $p \leq k$. For each R_i, we choose $u_i \in A^*(r)$, where $r \in R_i$. Then $A' = \cup_{i=1}^{p}\{(u_i, r) : r \in R_i\}$.

By definition, A' is complete. By (2) and (3), $A' \subseteq A^*$ implies $\Omega(A') \leq \Omega(A^*)$. Let $c = (r, r', \leftrightarrow, \forall) \in C$. If A^* satisfies c then A' also satisfies c. If c is falsified by A^* then

$$\max\{|A^*(r) \setminus A^*(r')|, |A^*(r') \setminus A^*(r)|\} \geq 1$$

but $\max\{|A'(r) \setminus A'(r')|, |A'(r') \setminus A'(r)|\} = 1$. Hence, $w_c(A^*) \geq f_c(1) = w_c(A')$. Now let $c = (r, r', \updownarrow, \forall) \in C$. By the proof of Lemma 5.1, we have $w_c(A^*) \geq w_c(A')$.

Thus, $\Omega(A') + w_C(A') \leq \Omega(A^*) + w_C(A^*)$. Since A^* is optimal, A' is optimal, too.

An instance of $\mathcal{I}'$ of Valued WSP(=, ≠) is defined as in Lemma 5.1, but the constraints $(r, r', =)$ correspond to constraints $(r, r', \leftrightarrow, \forall)$ in C. It is easy to see that the optimal solution of $\mathcal{I}$ has the same weight as that of $\mathcal{I}'$. ◻

We are now able to state the main result of this section. The result improves considerably on the running time for an algorithm that solves Valued APEP for arbitrary weighted t-bounded user-independent constraints (established in Theorem 4.14).

Theorem 5.3. Valued APEP$\langle \mathrm{BoD_U}, \mathrm{SoD_U}\rangle$ *is FPT and can be solved in time $O^*(2^{k \log k})$.*

Proof. Let $\mathcal{I}$ be an instance of Valued APEP$\langle \mathrm{BoD_U}, \mathrm{SoD_U}\rangle$. By Lemma 5.2, $\mathcal{I}$ can be reduced to an instance $\mathcal{I}'$ of Valued WSP(=, ≠). It remains to observe that $\mathcal{I}'$ can be solved in time $O^*(2^{k \log k})$ using the algorithm of Theorem 3.2, as $(r, r', =)$ and $(r, r', \neq)$ are user-independent constraints. ◻

6 $\mathrm{BoD_E}$ AND $\mathrm{SoD_U}$ CONSTRAINTS

In this section, we consider Valued APEP$\langle \mathrm{BoD_E}, \mathrm{SoD_U}\rangle$. We provide a construction that enables us to reduce an instance $\mathcal{I}$ of Valued APEP$\langle \mathrm{BoD_E}, \mathrm{SoD_U}\rangle$ with k resources to an instance $\mathcal{I}'$ of Valued WSP with only user-independent constraints containing at most $k(k-1)$ steps. Moreover, the construction yields a Valued WSP instance in which the weight of an optimal plan is equal to the weight of an optimal solution for the Valued APEP instance. Finally, we show that it is possible to construct the optimal solution for the VALUED APEP instance from an optimal plan for the VALUED WSP instance.

Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of VALUED APEP$\langle\mathrm{BoD_E}, \mathrm{SoD_U}\rangle$. (The weights of these types of constraints are defined by equations (11) and (8) in Section 2.1.) Let $R = \{r_1, \ldots, r_k\}$. Then we construct an instance $\mathcal{I}' = (S', U', C', \omega')$ of VALUED WSP as follows.

- Set $U' = U$.
- For every $i \in [k]$, first initialize a set $\Gamma(r_i) = \emptyset$. Then, for every $\mathrm{BoD_E}$ constraint $(r_i, r_j, \leftrightarrow, \exists) \in C$, add r_j to $\Gamma(r_i)$ and r_i to $\Gamma(r_j)$.
- For each resource $r_i \in R$, we create a set of steps $S' = \bigcup_{i=1}^{k} S^i$ where

$$S^i = \begin{cases} \{s^i\} & \text{if } \Gamma(r_i) = \emptyset, \\ \{s^i_j \mid r_j \in \Gamma(r_i)\} & \text{otherwise.} \end{cases}$$

Observe that $|S'| \leq k(k-1)$. Given a plan $\pi : S' \to U'$, we write $\pi(S^i)$ to denote $\{\pi(s) \mid s \in S^i\}$. Let Π be the set of all possible complete plans from S' to U'.

We define the set of constraints C' and their weights $w'_{c'} : \Pi \to \mathbb{N}$ as follows.

- For each $c = (r_i, r_j, \leftrightarrow, \exists) \in C$, we add constraint $c' = (s^i_j, s^j_i, =)$ to C', and define

$$w'_{c'}(\pi) = \begin{cases} 0 & \text{if } \pi(s^i_j) = \pi(s^j_i), \\ \ell_c & \text{otherwise.} \end{cases}$$

  Note that c' is user-independent.
- For each $c = (r_i, r_j, \updownarrow, \forall) \in C$, we add constraint $c' = (S^i, S^j, \emptyset)$, where c' is satisfied iff $\pi(S^i) \cap \pi(S^j) = \emptyset$. Then define $w'_{c'}(\pi) = f_c(|\pi(S^i) \cap \pi(S^j)|)$, where f_c is the function associated with the weighted constraint $(r_i, r_j, \updownarrow, \forall)$. Observe that $(S^i, S^j, \emptyset)$ is a user-independent constraint.
- Let C' denote the set of all constraints in $\mathcal{I}'$ and define

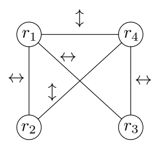
$$w'_{C'}(\pi) = \sum_{c' \in C'} w'_{c'}(\pi).$$

We then define authorization weight function $\omega' : U' \times 2^{S'} \to \mathbb{N}$ as follows. Initialize $\omega'(u, \emptyset) = 0$. We set $\omega'(u, S^i) = \omega(u, \{r_i\})$. For a subset $T \subseteq S'$, let $R_T = \{r_i \in R \mid T \cap S^i \neq \emptyset\}$. We set $\omega'(u, T) = \omega(u, R_T)$. Given a plan $\pi : S' \to U'$, we denote $\sum_{u \in U'} \omega'(u, \pi^{-1}(u))$ by $\Omega'(\pi)$. Finally, define the weight of π to be $\Omega'(\pi) + w'_{C'}(\pi)$. See Example 2 for an illustration.

Based on the construction described above, we have the following lemma:

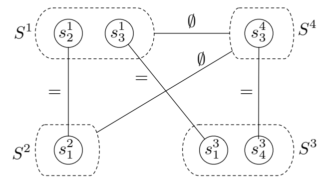
Lemma 6.1. *Let $\mathcal{I}$ be a* VALUED APEP$\langle\mathrm{BoD_E}, \mathrm{SoD_U}\rangle$ *instance and $\mathcal{I}'$ be the* VALUED WSP *instance obtained from $\mathcal{I}$ using the construction above. Then* $\mathrm{OPT}(\mathcal{I}) = \mathrm{OPT}(\mathcal{I}')$, *where* $\mathrm{OPT}(\mathcal{I})$ *and* $\mathrm{OPT}(\mathcal{I}')$ *denote the weights of optimal solutions of $\mathcal{I}$ and $\mathcal{I}'$ respectively. Furthermore, given an optimal plan for $\mathcal{I}'$, we can construct an optimal authorization relation for $\mathcal{I}$ in polynomial time.*

PROOF. We first prove that $\mathrm{OPT}(\mathcal{I}) \leq \mathrm{OPT}(\mathcal{I}')$. Let $\pi : S' \to U'$ be an optimal plan for the instance $\mathcal{I}'$. We construct A for the instance $\mathcal{I}$ as follows. For all $i \in [k]$, if $u \in \pi(S^i)$, then we put (u, r_i) into A. This completes the construction of A from π. Since π is complete, A is also complete. This can be implemented in polynomial time. Observe that $r_i \in A(u)$ if and only if there exists $s \in S^i$ such that $s \in \pi^{-1}(u)$. Equivalently, suppose that $T = \pi^{-1}(u)$. Then, $R_T = A(u)$. It



ω	r_1	r_2	r_3	r_4
u_1	0	0	1	1
u_2	0	1	0	0
u_3	1	0	1	1
u_4	1	1	0	1

(a) Valued APEP instance



ω'	S^1	S^2	S^3	S^4
u_1	0	0	1	1
u_2	0	1	0	0
u_3	1	0	1	1
u_4	1	1	0	1

(b) Valued APEP instance

Fig. 2. An example illustrating the construction. Note that $R = \{r_1, r_2, r_3, r_4\}$ and $S' = S^1 \cup S^2 \cup S^3 \cup S^4$.

implies that $\omega'(u, \pi^{-1}(u)) = \omega(u, A(u))$. Hence, we have

$$\Omega'(\pi) = \sum_{u \in U'} \omega'(u, \pi^{-1}(u)) = \sum_{u \in U} \omega(u, A(u)) = \Omega(A)$$

We now prove that $w_C(A) \leq w'_{C'}(\pi)$. Consider a BoD$_\mathsf{E}$ constraint $c = (r_i, r_j, \leftrightarrow, \exists) \in C$. Then, $r_j \in \Gamma(r_i)$, and $r_i \in \Gamma(r_j)$, and the corresponding constraint in $\mathcal{I}'$ is $c' = (s^i_j, s^j_i, =)$. By construction if $\pi(s^i_j) = \pi(s^j_i)$, then there exists $u \in A(r_i) \cap A(r_j)$. Hence, $w_c(A) \leq w'_{c'}(\pi)$. Now consider an SoD$_\mathsf{U}$ constraint $c = (r_i, r_j, \updownarrow, \forall)$. The corresponding constraint in $\mathcal{I}'$ is $c' = (S^i, S^j, \emptyset)$. Observe that by construction, if $\pi(S^i) \cap \pi(S^j) = \emptyset$, then $A(r_i) \cap A(r_j) = \emptyset$. Otherwise, if $|\pi(S^i) \cap \pi(S^j)| = t > 0$, then by construction $|A(r_i) \cap A(r_j)| = t$. Hence, $w'_{c'}(\pi) = w_c(A)$. We obtain an authorization relation A in polynomial time such that $w_C(A) + \Omega(A) \leq w'_{C'}(\pi) + \Omega'(\pi) = \mathrm{OPT}(\mathcal{I}')$. Therefore, $\mathrm{OPT}(\mathcal{I}) \leq \mathrm{OPT}(\mathcal{I}')$.

To complete the proof we prove that $\mathrm{OPT}(\mathcal{I}) \geq \mathrm{OPT}(\mathcal{I}')$. Let A be an optimal authorization relation for $\mathcal{I}$. We construct $\pi : S' \to U'$ as follows. If $\Gamma(r_i) = \emptyset$, we choose an arbitrary $u \in A(r_i)$ and set $\pi(s^i) = u$. Otherwise, $\Gamma(r_i) \neq \emptyset$, and two cases may arise.

- For $r_j \in \Gamma(r_i)$, $(r_i, r_j, \leftrightarrow, \exists)$ is satisfied by A. Then, we choose an arbitrary $u \in A(r_i) \cap A(r_j)$ and set $\pi(s^i_j) = \pi(s^j_i) = u$.
- For $r_j \in \Gamma(r_i)$, $(r_i, r_j, \leftrightarrow, \exists)$ is not satisfied by A. Then, we just choose arbitrary $u \in A(r_i), v \in A(r_j)$ and set $\pi(s^i_j) = u$, and $\pi(s^j_i) = v$.

This completes the construction of π. Note that π is complete.

Let $T = \pi^{-1}(u)$. Observe that by construction, if $u \in \pi(S^i)$, then $u \in A(r_i)$. Equivalently, if there exists $i \in [k]$ such that $\pi^{-1}(u) \cap S^i \neq \emptyset$, then $r_i \in A(u)$. Therefore, $R_T \subseteq A(u)$. Using the monotonicity property of ω, we have that $\omega(u, R_T) \leq \omega(u, A(u))$. This means that $\omega'(u, \pi^{-1}(u)) = \omega(u, R_T) \leq \omega(u, A(u))$. Therefore, we have the following:

$$\Omega'(\pi) = \sum_{u \in U'} \omega'(u, \pi^{-1}(u)) \leq \sum_{u \in U} \omega(u, A(u)) = \Omega(A)$$

Hence, $\Omega'(\pi) \leq \Omega(A)$.

Consider a BoD$_\mathsf{E}$ constraint $c = (r_i, r_j, \leftrightarrow, \exists) \in C$. By construction, c is satisfied by A if and only if $c' = (s^i_j, s^j_i, =)$ is satisfied by π. Hence, $w_{c'}(\pi) = w_c(A)$. On the other hand, consider an SoD$_\mathsf{U}$ constraint $c = (r_i, r_j, \updownarrow, \forall) \in C$. If c is satisfied by A, then by construction $c' = (S^i, S^j, \emptyset)$ is also satisfied by A. Finally, if c is violated by A, then let $t = |A(r_i) \cap A(r_j)| > 0$. By construction, $\pi(S^i) \cap \pi(S^j) \subseteq A(r_i) \cap A(r_j)$. Hence, $w'_{c'}(\pi) \leq w_c(A)$, implying $w'_{C'}(\pi) \leq w_C(A)$. Therefore, $\mathrm{OPT}(\mathcal{I}') \leq \mathrm{OPT}(\mathcal{I})$. □

Example 2. We illustrate the construction of a Valued WSP instance from a Valued APEP instance, and the proof of Lemma 6.1 using Figure 2. In addition, given an optimal solution for the corresponding Valued WSP instance, we illustrate how to construct an optimal solution of the Valued APEP instance as described in Lemma 6.1. As per the figure, there are three BoD_E constraints ($c_1 = (r_1, r_2, \leftrightarrow, \exists)$, $c_2 = (r_1, r_3, \leftrightarrow, \exists)$, and $c_3 = (r_3, r_4, \leftrightarrow, \exists)$) and two SoD_U constraints ($c_4 = (r_1, r_4, \updownarrow, \forall)$, and $c_5 = (r_2, r_4, \updownarrow, \forall)$). We define their weights as follows. For an authorization relation $A \subseteq U \times R$ and $i \in \{1, 2, 3\}$, we set $w_{c_i}(A) = 0$ if A satisfies c_i, and $w_{c_i}(A) = 1$, otherwise. Let $w_{c_4}(A) = 0$ if A satisfies c_4; otherwise, $w_{c_4}(A) = |A(r_1) \cap A(r_4)|$. Similarly, $w_{c_5}(A) = 0$ if A satisfies c_5; otherwise, $w_{c_5}(A) = |A(r_2) \cap A(r_4)|$. For every $\emptyset \neq T \subseteq R$ and $u \in U$, let $\omega(u, T) = \sum_{r \in T} \omega(u, \{r\})$.

Observe that in this example there is no authorization relation A such that $w_C(A) + \Omega(A) = 0$. It means that if we look for an authorization relation A such that $w_C(A) = 0$, then we will have $\Omega(A) > 0$. Consider an authorization relation A^* such that $A^*(r_1) = \{u_1, u_2\}, A^*(r_2) = \{u_1, u_3\}, A^*(r_3) = u_2$, and $A^*(r_4) = \{u_4\}$. Observe that $w_C(A^*) = 0$ but $\Omega(A^*) = 1$. Conversely, if we look for an authorization relation A such that $\Omega(A) = 0$, then we will have $w_C(A) > 0$.

Consider the Valued WSP instance constructed in this example. Based on the construction, $c'_1 = (s^1_2, s^2_1, =), c'_2 = (s^1_3, s^3_1, =), c'_3 = (s^3_4, s^4_3, =), c'_4 = (S^1, S^4, \emptyset)$, and $c'_5 = (S^2, S^4, \emptyset)$. Observe that for a given plan $\pi : S' \to U'$, we have the following:

- $w_{c'_1}(\pi) = 0$ if π satisfies c'_1 and $w_{c'_1}(\pi) = 1$ otherwise,
- $w_{c'_2}(\pi) = 0$ if π satisfies c'_2 and $w_{c'_2}(\pi) = 1$ otherwise,
- $w_{c'_3}(\pi) = 0$ if π satisfies c'_3 and $w_{c'_3}(\pi) = 1$ otherwise,
- $w_{c'_4}(\pi) = 0$ if π satisfies c'_4 and $w_{c'_4}(\pi) = |\pi(S^1) \cap \pi(S^4)|$ otherwise, and
- $w_{c'_5}(\pi) = 0$ if π satisfies c'_5 and $w_{c'_5}(\pi) = |\pi(S^2) \cap \pi(S^4)|$ otherwise.

Consider an optimal plan $\pi : S' \to U'$ defined as follows: $\pi(s^1_2) = u_1$, $\pi(s^1_3) = u_2$, $\pi(s^2_1) = u_1$, $\pi(s^3_1) = u_2$, $\pi(s^3_4) = u_4$, and $\pi(s^4_3) = u_4$. Observe that $w'_{C'}(\pi) = 0$ as all constraints c'_1, c'_2, c'_3, c'_4, and c'_5 are satisfied. Then, $\Omega'(u_1, \{s^1_2, s^2_1\}) = \Omega(u_1, \{r_1, r_2\}) = 0$, $\Omega'(u_2, \{s^1_3, s^3_1\}) = \Omega(u_2, \{r_1, r_3\}) = 0$, and $\Omega'(u_4, \{s^3_4, s^4_3\}) = \Omega(u_4, \{r_3, r_4\}) = 1$. Finally, $\Omega'(u_3, \emptyset) = 0$. Thus, $\Omega'(\pi) = 1$.

We construct A from π as in the first part of the the proof of Lemma 6.1: $A(r_1) = \{u_1, u_2\}, A(r_2) = \{u_1\}, A(r_3) = \{u_2, u_4\}$, and $A(r_4) = \{u_4\}$. Observe that $w_C(A) = 0$ as all constraints $c_1, \dots, c_5$ are satisfied by A and $\Omega(A) = 1$.

We can now state the main result of this section.

Theorem 6.2. Valued APEP$\langle \text{BoD}_\text{E}, \text{SoD}_\text{U} \rangle$ *is fixed-parameter tractable and can be solved in* $O^*(4^{k^2 \log k})$ *time.*

Proof. Let $\mathcal{I} = (R, U, C, \omega)$ be an instance of Valued APEP$\langle \text{BoD}_\text{E}, \text{SoD}_\text{U} \rangle$. We construct an instance $\mathcal{I}' = (S', U, C', \omega')$ of Valued WSP in polynomial time. We then invoke Theorem 3.2 to obtain an optimal plan $\pi : S' \to U'$. Finally, we invoke Lemma 6.1 to construct an optimal authorization relation A for $\mathcal{I}$ such that $\Omega(A) + w_C(A) = \text{OPT}(\mathcal{I}')$. The algorithm described in Theorem 3.2 runs in $O^*(2^{|S'| \log |S'|})$ time. Since $|S'| \leq k(k-1)$, the running time of this algorithm to solve Valued APEP$\langle \text{BoD}_\text{E}, \text{SoD}_\text{U} \rangle$ is $O^*(4^{k^2 \log k})$. □

7 USING VALUED APEP TO ADDRESS RESILIENCY IN WORKFLOWS

Resiliency, in the context of access control, is a generic term for the ability of an organization to continue to conduct business operations even when some authorized users are unavailable [24]. Resiliency is particularly interesting when an organization specifies authorization policies and separation of duty constraints, as is common in workflow systems, as separation of duty constraints become harder to satisfy when fewer (authorized) users are available.

Early work by Wang and Li showed that determining whether a workflow specification is resilient is a hard problem [24]. More recent work has established the precise complexity of determining static resiliency [18], and that the problem is FPT, provided all constraints in the workflow specification are user-independent [13].

We introduce the idea of an extended plan for a workflow specification and define resiliency in the context of an extended plan. We then explain how Valued APEP can be used to compute extended plans of minimal cost. In Section 8, we provide two MIP formulations for Valued APEP. Then in Section 9 we discuss our experimental framework and results making use of these two formulations.

Suppose we are given a workflow specification (defined by a set of workflow steps S, a set of users U, an authorization relation $\hat{A} \subseteq U \times S$ and a set of constraints C) and an integer $\tau \geq 0$. We call a function $\Pi : S \to 2^U$ an *extended plan*; we say Π is *valid* if there exists a valid plan $\pi : S \to U$ such that $\pi(s) \in \Pi(s)$ for all $s \in S$ and π is valid. We say Π is *τ-resilient* if for any subset of τ users $T \subseteq U$, there exists a valid plan $\pi' : S \to (U \setminus T)$ such that $\pi'(s) \in \Pi(s)$ for all $s \in S$. Wang and Li introduced an alternative notion of resiliency in workflows [30], where a workflow specification is said to be *statically t-resilient* if for all $U' \subseteq U$ such that $|U| - |U'| \leq t$, $(S, U', C, \hat{A}')$, where $\hat{A}' = \hat{A} \cap (S \times U')$, is satisfiable.

The two notions of resiliency are rather different. Our notion requires an extended plan to be resilient, so that having committed to an extended plan for a workflow we know the instance can complete even if τ users are unavailable. In contrast, Wang and Li require that the workflow specification itself is resilient. Crampton, Gutin, Karapetyan and Watrigant showed that determining whether a workflow is statically t-resilient is FPT [13] for WSP with UI constraints only.

It is not obvious that the methods used by Crampton *et al.* can be adapted to determine whether there exists a τ-resilient extended plan. Nor is it obvious whether the problem of deciding if there exists a τ-resilient extended plan can be framed as an instance of APEP.

APEP, however, can be used to produce an extended plan that is τ-resilient. Moreover, Valued APEP can be used to solve the softer problem of finding an extended plan that aims to be τ-resilient (but may not be) and that also minimises the number of users involved.

To generate a τ-resilient extended plan for a WSP instance $(S, U, C, \hat{A})$ with SoD constraints, we can produce the following APEP instance $(R', U', C', \hat{A}')$:

- Let $R' = S$, $U' = U$ and $\hat{A}' = \hat{A}$.
- Let $C' = \emptyset$. For every $c \in C$, add a corresponding SoD_U to C' (recall that we consider WSP with SoD constraints only).
- Add Cardinality-Lower-Bound constraints $(r, \geq, \tau + 1)$ for every $r \in R$.

Any authorization relation A that satisfies such an APEP instance is a τ-resilient extended plan in the original WSP instance. (Note that it is sufficient for an extended plan Π to be a solution of an APEP instance, however it is not necessary; some τ-resilient extended plans may not be solutions for an APEP instance.)

The requirement to have at least $\tau + 1$ users assigned to each resource may lead to solutions that involve too many users. In practice, we may want to keep the number of users involved in Π as small as possible. Also, where an instance is not τ-resilient, we may want to accept solutions that are not completely τ-resilient, i.e. solutions where excluding τ users may render the extended plan invalid to some (acceptably limited) extent.

To meet the above requirements, we can use the Valued APEP to model τ-resiliency in WSP. Let p_{SoD} and p_{Card} be penalties for violation of the corresponding constraints. Let p_A be a penalty for assigning a user to a resource for which this user is not authorized. Compose a APEP instance $(R, U, C, \hat{A})$ as described above and replace each constraint with the following weighted constraints:

- For every SoD_U constraint $c = (r_1, r_2, \updownarrow, \forall)$, let $w_c(A) = p_{\mathrm{SoD}} \cdot |A(r_1) \cap A(r_2)|$; i.e., there is a p_{SoD} penalty for every user assigned to both resources in the scope.
- For every cardinality-lower-bound constraint $c = (r, \geq, \tau + 1)$, let $w_c = \max\{0, p_{\mathrm{Card}} \cdot (\tau + 1 - |A(r)|)\}$.
- Finally, we add a constraint c, which we call *User Count Constraint*, with the scope R such that $w_c = f_\Pi(|A(R)|)$, where $f_\Pi(\cdot)$ is a monotonically growing function. Specifically, we use $w_c = |A(R)|^2$.

Also let $\omega(u, T) = p_A \cdot \ell$, where ℓ is the number of resources $r \in T$ such that $(r, u) \notin \hat{A}$. In other words, there is a p_A penalty for each unauthorized assignment of a user to a resource.

Assuming p_A, p_{SoD} and p_{Card} are positive numbers and the original APEP instance is satisfiable, a solution to this Valued APEP instance will be a τ-resilient extended plan with at least $\tau + 1$ users assigned to each resource, with all the SoD_U constraints and authorizations satisfied and with the number of users involved in the extended plan minimized.

8 MIXED INTEGER FORMULATIONS OF VALUED APEP

While it is common to implement bespoke algorithms to exploit the FPT properties of a problem, it was noted recently that off-the-shelf solvers may also be efficient on such problems given appropriate formulations [21, 22]. In this section we give two mixed integer programming (MIP) formulations of the Valued APEP. The formulation given in Section 8.1 is a straightforward interpretation of the problem; it uses binary variables to define an assignment of users to resources. The formulation given in Section 8.2, however, makes use of the concept of user profiles, used earlier to prove FPT results. While both formulations are generic enough to support any Valued APEP constraints, we focus on the constraints used to model τ-resiliency of WSP extended plans, see Section 7.

8.1 Naive formulation

The *Naive* formulation of a Valued APEP instance (R, U, C, ω) is based on binary variables $x_{r,u}$ linking resources to users; $x_{r,u} = 1$ if and only if user u is assigned to resource r.

The core of the formulation is as follows:

$$\text{minimize} \sum_{c \in C} p_c + p_A \cdot \sum_{(r,u) \notin \hat{A}} x_{r,u} \tag{12}$$

subject to

$$x_{r,u} \in \{0, 1\} \qquad \forall r \in R,\ \forall u \in U, \tag{13}$$

$$p_c \in [0, \infty] \qquad \forall c \in C. \tag{14}$$

The encodings of the Valued APEP constraints linking the solution to variables p_c are discussed below.

The User Count constraint c is encoded as follows:

$$p_c = f_\Pi(z), \tag{15}$$

$$z = \sum_{u \in U} y_u, \tag{16}$$

$$y_u \geq x_{r,u} \qquad \forall r \in R,\ \forall u \in U, \tag{17}$$

$$y_u \in [0, 1] \qquad \forall u \in U, \tag{18}$$

$$z \in [0, n]. \tag{19}$$

Variable z is introduced to count the number $|A(R)|$ of users generated by the solution. The formulation depends on the function f_Π; for $f_\Pi(z) = z^2$, we use the following encoding as a discretization of a parabola:

$$p_c \geq f_i(z) \qquad \forall i \in \{1, 2, \ldots, n-1\}, \tag{20}$$

where $f_i(z) = (2i+1)z - (i+1)i$. An illustration of how it enforces $p_c \geq z^2$ is given in Figure 3. Note that $z^2 \geq f_i(z)$ for every integer z and i.

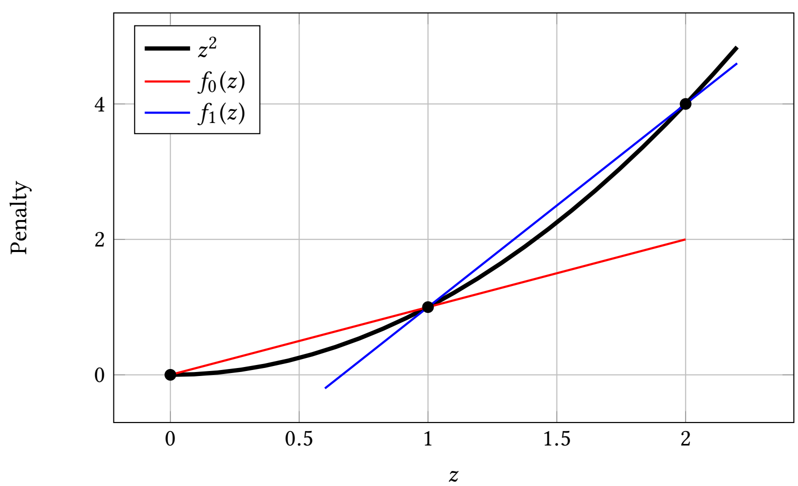


Fig. 3. Illustration of how equations (20) enforce $p_c \geq z^2$.

Each Cardinality-Lower-Bound constraint $c = (r, \geq, \tau + 1)$ is encoded as follows:

$$p_c \geq p_{\text{Card}} \cdot \left[(\tau + 1) - \sum_{u \in U} x_{r,u} \right]. \tag{21}$$

Each SoD constraint $c = (r_1, r_2, \updownarrow, \forall)$ is encoded as follows:

$$p_c = p_{\text{SoD}} \cdot \sum_{u \in U} y_u, \tag{22}$$

$$y_u >= x_{r_1,u} + x_{r_2,u} - 1 \qquad \forall u \in U, \tag{23}$$

$$y_u \in \{0, 1\} \qquad \forall u \in U. \tag{24}$$

8.2 User-Profile formulation

The *User-Profile* (UP) formulation is based on the concept of user profiles making it an FPT-aware formulation. Let $\mathcal{T}$ be the power set of R. The UP formulation defines a binary variable $x_{T,u}$ for every $T \in \mathcal{T}$ and $u \in U$. We then require that each user u is assigned exactly one $T \in \mathcal{T}$.

The core of the formulation is as follows:

$$\text{minimize} \sum_{c \in C} p_c + p_A \cdot \sum_{T \in \mathcal{T}} \sum_{u \in U} x_{T,u} \cdot |\{r \in T : (r,u) \notin \hat{A}\}| \tag{25}$$

subject to

$$x_{T,u} \in \{0,1\} \qquad \forall T \in \mathcal{T},\ \forall u \in U, \tag{26}$$

$$p_c \in [0, \infty] \qquad \forall c \in C. \tag{27}$$

The encodings of the Valued APEP constraints linking the solution to variables p_c are discussed below.

The encoding of the User Count constraint c is very similar to its implementation in the Naive formulation:

$$p_c = f_\Pi(z), \tag{28}$$

$$z = \sum_{u \in U} y_u, \tag{29}$$

$$y_u \geq x_{T,u} \qquad \forall T \in \mathcal{T} \setminus \{\emptyset\},\ \forall u \in U, \tag{30}$$

$$y_u \in [0,1] \qquad \forall u \in U, \tag{31}$$

$$z \in [0, n]. \tag{32}$$

Specifically, we use $f_\Pi(z) = z^2$, which we encode as follows (see (20) for details):

$$p_c \geq f_i(z) \qquad \forall i \in \{1, 2, \dots, n-1\}. \tag{33}$$

Each Cardinality-Lower-Bound constraint $c = (r, \geq, \tau + 1)$ is encoded as follows:

$$p_c \geq p_{\text{Card}} \cdot \left[(\tau + 1) - \sum_{T \in \mathcal{T},\, r \in T} \sum_{u \in U} x_{T,u} \right]. \tag{34}$$

Each SoD constraint $c = (r_1, r_2, \updownarrow, \forall)$ is encoded as follows:

$$p_c = p_{\text{SoD}} \cdot \sum_{T \in \mathcal{T},\, r_1, r_2 \in T} \sum_{u \in U} x_{T,u}. \tag{35}$$

9 COMPUTATIONAL EXPERIMENTS

The aims of our computational study are to:

(1) design an instance generator, to support future experimental studies of the Valued APEP and enable fair comparison of Valued APEP solution methods;
(2) give a new approach to address resiliency in WSP;
(3) compare the performance of the two formulations discussed in Section 8;
(4) test if either of the formulations has FPT-like running times, i.e. scales polynomially with the instance size given that the small parameter is fixed;
(5) analyse the structure of optimal solutions and how it depends on the instance generator inputs; and
(6) make the instance generator and the solvers based on the two formulations publicly available.

9.1 Benchmark instances

For our computational experiments, we built a pseudo-random instance generator for the Valued APEP. It is designed around the concept of τ-resiliency of WSP, see Section 7. The inputs of the instance generator are detailed in Table 2.

Input	Description	Default value
n	the number of users, also referred to as the size of the problem	–
k	the number of WSP steps	$\lfloor 0.1n \rfloor$
τ	the desired degree of τ-resiliency, i.e. the number of users that can be excluded from the extended plan	$\lfloor 0.05n \rfloor$
α	an input enabling us to adjust the balance between the penalties associated with workflow and resiliency violations	1

Table 2. The instance generator inputs and their default values.

While inputs n and k define the size of the instance and τ is an inherent input of τ-resiliency, α is an artifact of the experimental set-up, introduced to control the weight of the workflow constraints and authorizations relative to resiliency. Varying the value of α enables us to investigate the effects of emphasizing the importance of satisfying workflow constraints (over resiliency) and vice versa. The greater the value of α, the greater the penalties for violating workflow constraints and authorizations, meaning that satisfying resiliency becomes correspondingly less significant.

The instance generator first creates a WSP instance $(S, U, C, \hat{A})$ and then converts that instance into a Valued APEP instance (R, U, C', ω), as described in Section 7. The constraint penalties are set as following: $p_{\text{SoD}} = 10\alpha$, $p_{\text{Card}} = 10$ and $p_A = \alpha$. The WSP instance is generated in the following way:

(1) create steps $S = \{s_1, s_2, \ldots, s_k\}$ and users $U = \{u_1, u_2, \ldots, u_n\}$;
(2) the authorizations are created in the same way as in the WSP instance generator, see [22]: for each user $u \in U$, select randomly and uniformly from $[1, \lfloor 0.5 \cdot (k-1) \rfloor]$ the number of steps for which u is authorized, and then randomly select which steps they are authorized to; and
(3) produces q_{SoD} constraints SoD, selecting the scope of each of them randomly and independently (the generator may produce several SoD constraints with the same scope).

9.2 t-wboundness of the User Count constraint

It is a trivial observation that the User Count constraint is 0-wbounded. However, we can also establish the t-wboundness of a set of constraints C, where C includes a User Count constraint.

Proposition 9.1. *Let C_{SoD} be a set of SoD$_U$ constraints. Let C_{Card} be a set of Cardinality-Lower-Bound constraints with the penalty function $w_c(A) = \max\{0, p_{Card} \cdot (\ell - |A(r)|)\}$ for some ℓ. Let c_Π be a User Count constraint with the penalty function $w_{c_\Pi}(A) = |A(R)|^2$. Let $C = C_{SoD} \cup C_{Card} \cup \{c_\Pi\}$. Then C is $0.5k \cdot (|C_{Card}| \cdot p_{Card} + 1)$-bounded.*

Proof. Let us assume that A is an authorization relation that minimizes $w_C(A)$ and that $|A| > 0.5k \cdot (|C_{\text{Card}}| \cdot p_{\text{Card}} + 1)$. Observe that $|A(R)| > 0.5 \cdot (|C_{\text{Card}}| \cdot p_{\text{Card}} + 1)$. We will show a contradiction by constructing an authorization relation A' such that $w_C(A') < w_C(A)$ and $|A'(R)| = |A(R)| - 1$.

Select an arbitrary user $u \in U$ such that $A(u) \neq \emptyset$. Let A' be an authorization relation such that $A'(u') = A(u')$ for $u' \in U \setminus \{u\}$ and $A'(u) = \emptyset$. (Effectively, we exclude one user involved in the authorization relation.) Note that

(1) $w_c(A') - w_c(A) \leq 0$ for every $c \in C_{\mathrm{SoD}}$ as excluding a user cannot increase the SoD penalty.
(2) $w_c(A') - w_c(A) \leq p_{\mathrm{Card}}$ as the penalty of a Cardinality-Lower-Bound constraint can only increase by p_{Card} following an exclusion of a single user.
(3) $w_{c_\Pi}(A') - w_{c_\Pi}(A) = |A'(R)|^2 - |A(R)|^2 = (|A(R)| - 1)^2 - |A(R)|^2 = -2|A(R)| + 1$.

Hence, $w_C(A') - w_C(A) \leq |C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} - 2|A(R)| + 1$. Since $|A(R)| > 0.5 \cdot (|C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} + 1)$,

$$w_C(A') - w_C(A) < |C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} - 2 \cdot 0.5 \cdot (|C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} + 1) + 1 = 0.$$

In other words, $w_C(A') < w_C(A)$ which is a contradiction to the assumption that A minimizes $w_C(A)$. Hence, an optimal authorization relation A cannot be of size greater than $0.5k \cdot (|C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} - 1)$, i.e. C is $0.5k \cdot (|C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} + 1)$-wbounded. □

Proposition 9.1 is important because it shows that the instances produced by our instance generator are t-wbounded and that t does not depend on α, τ or n. Hence, an FPT algorithm is expected to scale polynomially with n if k is fixed, even though τ is a function of n. We will use this as a test for FPT-like running times.

9.3 Computational results

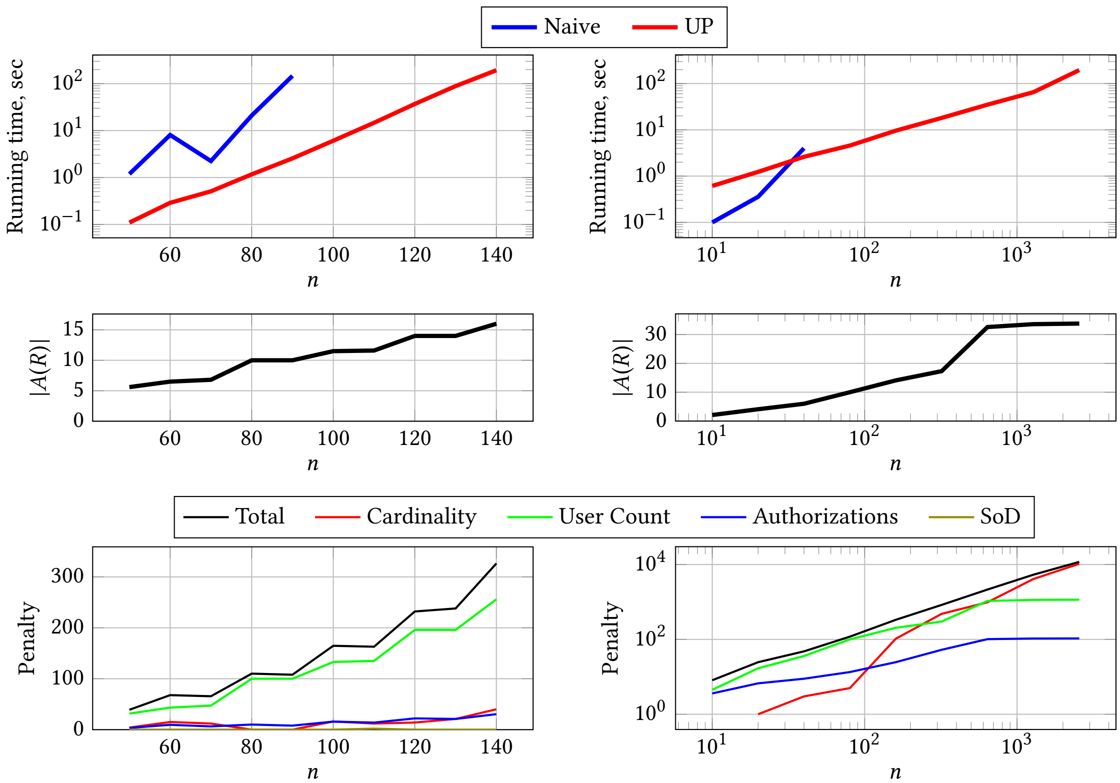
We used IBM CPLEX 20.1 to solve the MIP formulations. The formulations were generated using Python 3.8.8 scripts available at doi.org/10.17639/nott.7124. The experiments were conducted on a Dell XPS 15 9570 with Intel i7-8750H CPU (2.20 GHz) and 32 GB of RAM. CPLEX was allowed to use all the CPU cores. Only one instance of CPLEX would run at any point in time. Each experiment was repeated 10 times for 10 different instances produced with different random number generator seed values. The results reported in this section are the averages over the 10 runs.

9.3.1 Scaling. In our first set of experiments we adjust the instance size n and analyse how this affects the solution time and the optimal solution properties. This is particularly important to understand the limitations of the methods in terms of the instance size that they can handle, as well as study the structure of solutions to large instances. In Figure 4a, we change both k and n (and all the associated instance generator inputs), to test how the runtime of the solvers scale. However, as the problem is FPT, we also tested in Figure 4b how the solution time and the optimal solution properties change if the value of the small parameter k is fixed while the problem size n changes.

We notice that the UP solver generally outperforms the Naive solver by a large margin; in fact, it scales much better, hence the gap between the solvers increases with the instance size. The running time of the UP solver seems to be exponential only in k and linear in n; i.e., it has FPT-like running time. It is hard to determine how the Naive solver's running time scales as we could only obtain a few data points but it appears that its running time scales super-polynomially even if k is fixed meaning that its running time is not FPT-like. In other words, we believe that the UP solver efficiently exploits the FPT structure of the problem whereas the Naive solver fails to do so.

When we scale both k and n (Figure 4a), the number of users $|A(R)|$ in the optimal solutions grows linearly. However when we fix k (Figure 4b), there seems to be an upper bound on $|A(R)|$. This is consistent with our expectations; according to Proposition 9.1, the number of users is expected to be bounded by $0.5(|C_{\mathrm{Card}}| \cdot p_{\mathrm{Card}} + 1) = 0.5(10k + 1)$. For $k = 10$, this gives us an upper bound of around 50. The discrepancy with the practice is due to the influence of the SoD constraints and authorizations, both generating pressure to keep the number of users small.

As long as k is comparable to n, the User Count constraint is the main cause of the penalty. However, as n gets bigger relative to k, the cardinality constraints penalty begins to dominate. This is due to the relation between n and τ; while the number of users stays unchanged as we increase n, the value of τ grows as does the penalty caused by



(a) $k = \lfloor 0.1n \rfloor$.

(b) $k = 10$.

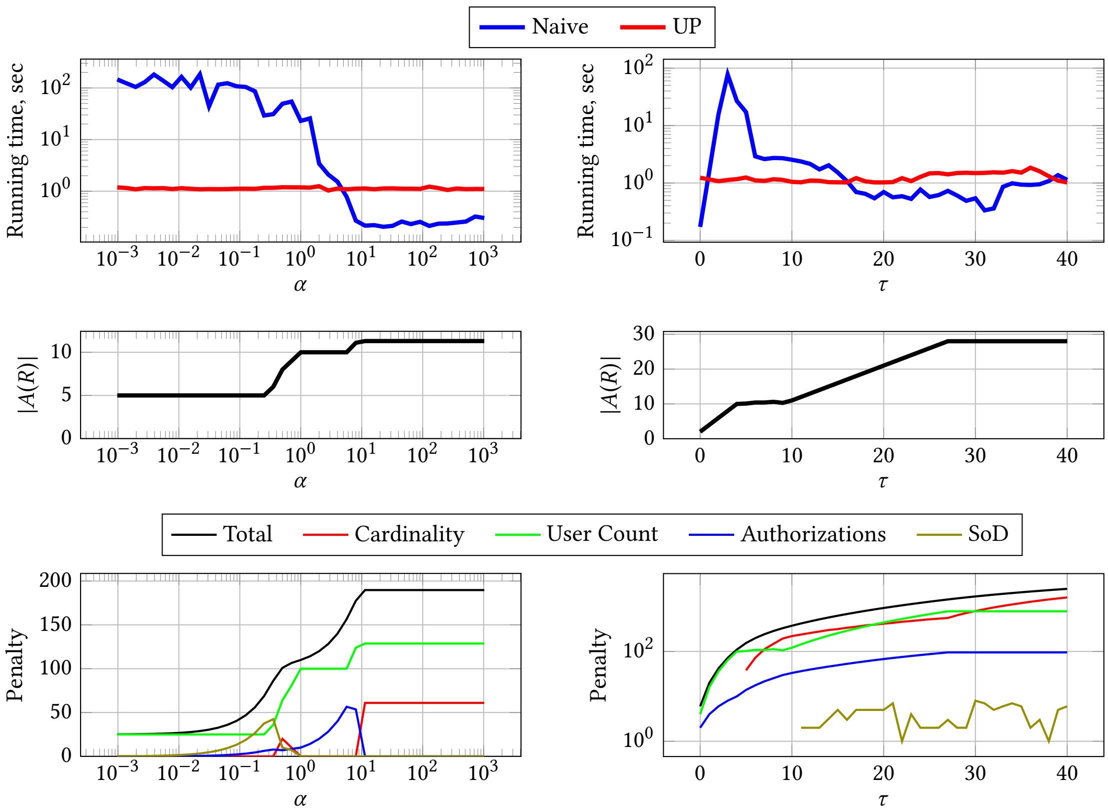
Fig. 4. Scaling of the solution time, number $|A(R)|$ of users involved in the solution and the penalties as the instance size changes. In all instances, $\alpha = 1$ and $\tau = \lfloor 0.05n \rfloor$.

violations of the cardinality constraints. With a few minor exceptions, all the SoD constraints are satisfied in all the experiments, whereas authorizations are often violated to a small extent. This imbalance is due to the 10-fold difference in the corresponding penalties.

9.3.2 Sensitivity to α and τ. The second set of experiments is designed to analyse the impact of the instance generator inputs α and τ on the instances, optimal solutions and the running times of the solvers. The results are presented in Figure 5.

These experiments reveal that the values of α and τ have little effect on the running time of UP. In fact, the running time is consistently proportional to the size of the formulation $O(n \cdot 2^k)$. Also, the composition of the formulation takes about half of the running time. In other words, CPLEX solves this formulation in time linear in its size but the size of the formulation is exponential in k putting a limit on how far this method can be scaled.

Thus, the Naive solver outperforms the UP solver in some extreme cases; when the instances are easy, the Naive formulation can exploit their special structure whereas the UP formulation remains large and as a result slow. For example, when α is large, breaking the SoD constraints and authorizations becomes prohibitively expensive which significantly reduces the search space for the Naive solver.



(a) Changing input α while $\tau = 4$.

(b) Changing input τ while $\alpha = 1$.

Fig. 5. Analysis of the solution time, number $|A(R)|$ of users involved in the solution and the penalties as the instance generator inputs change. In all instances, $k = 8$ and $n = 80$.

However, when α is close to 1 and τ is small, the instances are particularly challenging as the optimal solutions tend to balance the penalties of all types, and this is where UP is particularly effective compared to Naive.

9.3.3 Experiment conclusions. For an organization that implements a workflow management system and has strict business continuity requirements, it will be important to find a trade-off between satisfying authorization policies and constraints and ensuring that workflow instances can complete when users are unavailable. We believe these experiments provide some useful insights into the interplay between authorization policies, separation of duty constraints and resiliency, and form a basis from which costs of violating policies and resiliency can be balanced.

The trade-offs between authorization and resiliency requirements are evident in Figure 5a. For very small values of α (when penalties for violating authorization requirements are low) the penalties in Valued APEP solutions are dominated by $|A(R)|$, the number of users assigned to the extended plan. As α increases, the penalties associated with violations of authorization and constraints begin to dominate, and, as α increases further (meaning that authorizations and constraints become increasingly expensive to violate) the penalties associated with breaking resiliency and $|A(R)|$ dominate. We also see that $|A(R)|$ reaches a maximum value when α equals around 10, at which point the penalties associated with violating authorization requirements are negligible compared to those associated with resiliency and $|A(R)|$.

From Figure 5b we see that for $\tau \leq 4$ resiliency requirements are always satisfied. For $\tau > 4$, unsurprisingly, τ-resilient extended plans would have to be a large proportion of the user population and the penalties associated with such plans begin to dominate. For very large values of τ we find that $|A(R)|$ drops below the value of τ, meaning the extended plans cannot be τ-resilient and the penalties for such plans are mainly associated with resiliency and the size of the solution.

Finally, we note that the concept of user profiles, used to prove theoretical results about Valued APEP, also enables us to derive a MIP formulation that can be solved by CPLEX in FPT-like time and that is efficient across all our test instances. In contrast, the straightforward (Naive) formulation of the problem scales super-polynomially with the size of the problem even if the small parameter is fixed. This demonstrates the importance of FPT algorithms even if the researcher intends to use general-purpose solvers to address the problem.

10 RELATED WORK AND DISCUSSION

Valued APEP builds on a number of different strands of recent research in access control, including workflow satisfiability, workflow resiliency and risk-aware access control. Workflow satisfiability is concerned with finding an allocation of users to workflow steps such that every user is authorized for the steps to which they are assigned and all workflow constraints are satisfied. Work in this area began with the seminal paper by Bertino *et al.* [3]. Wang and Li initiated the use of parameterized complexity analysis to better understand workflow satisfiability [30], subsequently extended to include user-independent constraints [7].

Crampton et al. introduced Valued WSP [12] and Bi-objective WSP [13] in order to find plans for unsatisfiable instances of WSP: a cost is assigned to each constraint and assignment; the goal is to find a plan that minimizes the total cost of breaking constraints and authorizations. Inspired by [12, 13], Bertolissi et al. [5] studied Bi-objective Ordered Execution WSP, solutions for which also specify an ordering on the steps in the plan. While we use an MIP solver, Bertolissi et al. used Optimization Modulo Theories solvers. As we have seen APEP can be used to encode workflow satisfiability problems.

Basin et al. [1] consider the Optimal Workflow-Aware Authorization Administration Problem (OWA) – the problem of finding an optimal (minimal cost) authorization relation, given a workflow specification and a function that determines the (administrative cost) of changing the existing authorization relation to a different one. OWA could be used to solve simplified instances of Valued WSP, in the sense that it would be possible to use the returned authorization relation to find a valid plan. However, this approach does not allow for any breaking of constraints. Conversely, Valued WSP cannot directly solve OWA, as a solution for Valued WSP is a plan of minimal cost. Nor is it obvious that OWA could be treated as an instance of APEP: in particular, the objective in Valued APEP is to minimize the cost of policy violations (in OWA the objective is to minimize the cost of modifying one authorization relation to another); and it is not clear what the base authorization relation should be.

The papers [4, 16] consider the problem of computing what could informally be called potential plans for a workflow specification in which the sets of steps and constraints are given, but not the set of users or the authorization relation. Users are symbolic and all possible plans for this set of symbolic users are pre-computed. The idea is that the workflow specification will be used by many different customers who will use the pre-computed plans to determine whether there is a way of associating authorized users with symbolic users, given a customer's particular instantiation of the authorization relation. The customers may use symbolic model checking to find a valid plan. The techniques used to develop FPT algorithms for WSP [7, 10] could be used to construct similar graphs to those used in [4, 16].

Workflow resiliency is concerned with ensuring business continuity in the event that some (authorized) users are unavailable to perform steps in a workflow [18, 24, 26, 31]. Bergé et al. showed that APEP can be used to encode certain kinds of resiliency policies [2, Section 6]. In this paper, we introduce the notion of an extended plan and what it means for such a plan to be resilient. We believe this is a useful alternative to prior definitions of resiliency, in that there is no requirement for the workflow specification itself to be resilient. Thus, when an organization is aware of potential staff shortages, for example, it can require that a particular instance of a workflow is resilient. Moreover, Valued APEP allows the organization to trade the costs of resiliency and worfklow satisfaction when it is not possible to find a fully resilient extended plan. Researchers in access control have recognized that it may be necessary to violate access control policies in certain, exceptional circumstances [27, 28], provided that those violations are controlled appropriately. One means of controlling violations is by assigning a cost to policy violations, usually defined in terms of risk [6, 15]. Thus, the formalization of problems such as Valued WSP and Valued APEP and the development of algorithms to solve these problems may be of use in developing risk-aware access control systems.

Thus, we believe that APEP and Valued APEP are interesting and relevant problems, and understanding the complexity of these problems and developing the most efficient algorithms possible to solve them is important. A considerable amount of work has been done on the complexity of WSP, showing that the problem is FPT for many important classes of constraints [7, 11, 22]. It is also known that Valued WSP is FPT and, for user-independent constraints, the complexity of the problem is identical to that for WSP (when polynomial terms in the sizes of the user set and constraint set are disregarded in the running time) [12]. Roughly speaking, this is because (weighted) user-independent constraints in the context of workflow satisfiability allow us to restrict our attention to partitions of the set of steps when searching for solutions, giving rise to the exponential term $2^{k \log k}$ in the running time of an algorithm to solve (Valued) WSP.

APEP, unsurprisingly, is known to be a more complex problem [2]. The complexity of APEP differs from WSP because it is not sufficient to consider partitions of the set of resources, in part because an arbitrary relation A is not a function. The results in this paper provide the first complexity results for Valued APEP, showing (in Corollary 4.15) that it is no more difficult than APEP for constraints in $\mathrm{BoD_U}$, $\mathrm{BoD_E}$, $\mathrm{SoD_U}$ and $\mathrm{SoD_E}$ (disregarding polynomial terms).

We believe the concept of a user profile and Theorem 4.14 are important contributions to the study of APEP as well as Valued APEP, providing a generic way of establishing complexity results for different classes of constraints. In particular, Corollary 4.15 of Theorem 4.14 actually shows how to improve existing results for APEP$\langle \mathrm{BoD_U}, \mathrm{BoD_E}, \mathrm{SoD_E}, \mathrm{SoD_U} \rangle$ due to Bergé et al. [2]. Moreover, when an APEP instance is equivalent to a WSP instance (i.e, it contains a cardinality constraint $(r, \leq, 1)$ for each $r \in R$) then the instance is k-bounded, and a user profile is the characteristic function of some partition of R. Thus we essentially recover the known FPT result for Valued WSP, which is based on the enumeration of partitions of the set of workflow steps.

11 CONCLUDING REMARKS

We believe this paper makes three significant contributions. First, we introduce Valued APEP, a generalization of APEP, which, unlike APEP, always returns some authorization relation. Thus a solution to Valued APEP is more useful than that provided by APEP: if there exists a valid authorization relation Valued APEP will return it; if not, Valued APEP returns a solution of minimum weight. This allows an administrator, for example, to decide whether to implement the solution for an instance of Valued APEP or adjust the base authorization relation and/or the constraints in the input in an attempt to find a more appropriate solution.

The second contribution is to advance the techniques available for solving APEP as well as Valued APEP. Specifically, the notion of a user profile plays a similar role in the development of algorithms to solve (Valued) APEP as patterns do in solving (Valued) WSP. The enumeration of user profiles is a powerful technique for analyzing the complexity of Valued APEP, yielding general results for the complexity of the problem (which are optimal assuming the Exponential Time Hypothesis holds) and improved results for APEP.

The third contribution is the experimental study that involves a new set of realistic benchmark instances, two mixed integer programming formulations of Valued APEP and extensive analysis of the computational results. Apart from the conclusions related to the new concept of τ-resiliency in workflows, we demonstrate that a general-purpose solver can solve an FPT problem in FPT-like time if the formulation is 'FPT-aware': i.e., if it exploits our understanding of the FPT properties of the problem, and that such an 'FPT-aware' formulation significantly outperforms a naive formulation. This is particularly significant for practitioners who often prefer to use general purpose solvers to address complex problems, as they can now benefit from theoretical results in parameterized computational complexity.

There are several opportunities for further work. We intend to investigate other (weighted) user-independent constraints for (Valued) APEP. First, we are interested in what other problems in access control can be encoded as APEP instances, apart from workflow satisfiability and resiliency problems. Second, we would like to consider appropriate weight functions for such encodings, which would have the effect of providing more useful (weighted) solutions for the original problems (rather a binary yes/no solution). Our work also paves the way for work on quantifying the trade-offs associated with violating security and resiliency requirements when it is impossible to satisfy both simultaneously. A better understanding of these trade-offs together with tools for computing optimal solutions would seem to have considerable value to commercial organizations, enabling them to manage conflicting security and business requirements in an informed manner.

12 ACKNOWLEDGMENTS

The research of Crampton, Gutin and Majumdar was supported by the Leverhulme Trust award RPG-2018-161. We are very grateful to the reviewers for careful reading of the manuscript and giving us many helpful comments and suggestions.

REFERENCES

[1] David A. Basin, Samuel J. Burri, and Günter Karjoth. 2012. Optimal workflow-aware authorizations. In *17th ACM Symposium on Access Control Models and Technologies, SACMAT '12, Newark, NJ, USA - June 20 - 22, 2012*, Vijay Atluri, Jaideep Vaidya, Axel Kern, and Murat Kantarcioglu (Eds.). ACM, 93–102.

[2] P. Bergé, J. Crampton, G. Gutin, and R. Watrigant. 2020. The Authorization Policy Existence Problem. *IEEE Transactions on Dependable and Secure Computing* 17, 6 (2020), 1333–1344.

[3] Elisa Bertino, Elena Ferrari, and Vijayalakshmi Atluri. 1999. The Specification and Enforcement of Authorization Constraints in Workflow Management Systems. *ACM Trans. Inf. Syst. Secur.* 2, 1 (1999), 65–104.

[4] Clara Bertolissi, Daniel Ricardo dos Santos, and Silvio Ranise. 2015. Automated Synthesis of Run-time Monitors to Enforce Authorization Policies in Business Processes. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '15, Singapore, April 14-17, 2015*, Feng Bao, Steven Miller, Jianying Zhou, and Gail-Joon Ahn (Eds.). ACM, 297–308.

[5] Clara Bertolissi, Daniel Ricardo dos Santos, and Silvio Ranise. 2018. Solving Multi-Objective Workflow Satisfiability Problems with Optimization Modulo Theories Techniques. In *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies, SACMAT 2018, Indianapolis, IN, USA, June 13-15, 2018*, Elisa Bertino, Dan Lin, and Jorge Lobo (Eds.). ACM, 117–128.

[6] Liang Chen and Jason Crampton. 2011. Risk-Aware Role-Based Access Control. In *STM (Lecture Notes in Computer Science, Vol. 7170)*. Springer, 140–156.

[7] David Cohen, Jason Crampton, Andrei Gagarin, Gregory Gutin, and Mark Jones. 2014. Iterative Plan Construction for the Workflow Satisfiability Problem. *J. Artif. Intell. Res. (JAIR)* 51 (2014), 555–577. https://doi.org/10.1613/jair.4435

[8] David A. Cohen, Martin C. Cooper, Peter Jeavons, and Andrei A. Krokhin. 2005. Supermodular functions and the complexity of MAX CSP. *Discret. Appl. Math.* 149, 1-3 (2005), 53–72. https://doi.org/10.1016/j.dam.2005.03.003

[9] Jason Crampton, Eduard Eiben, Gregory Z. Gutin, Daniel Karapetyan, and Diptapriyo Majumdar. 2021. Valued Authorization Policy Existence Problem. In *SACMAT '21: The 26th ACM Symposium on Access Control Models and Technologies, Virtual Event, Spain, June 16-18, 2021*, Jorge Lobo, Roberto Di Pietro, Omar Chowdhury, and Hongxin Hu (Eds.). ACM, 83–94. https://doi.org/10.1145/3450569.3463571

[10] Jason Crampton, Gregory Gutin, and Rémi Watrigant. 2016. Resiliency Policies in Access Control Revisited. In *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies*. ACM, 101–111. https://doi.org/10.1145/2914642.2914650

[11] Jason Crampton, Gregory Gutin, and Anders Yeo. 2013. On the Parameterized Complexity and Kernelization of the Workflow Satisfiability Problem. *ACM Trans. Inf. Syst. Secur.* 16, 1 (2013), 4. https://doi.org/10.1145/2487222.2487226

[12] Jason Crampton, Gregory Z. Gutin, and Daniel Karapetyan. 2015. Valued Workflow Satisfiability Problem. In *SACMAT*. ACM, 3–13.

[13] Jason Crampton, Gregory Z. Gutin, Daniel Karapetyan, and Rémi Watrigant. 2017. The bi-objective workflow satisfiability problem and workflow resiliency. *J. Comput. Secur.* 25, 1 (2017), 83–115.

[14] M. Cygan, F.V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. 2015. *Parameterized Algorithms*. Springer.

[15] Nathan Dimmock, András Belokosztolszki, David M. Eyers, Jean Bacon, and Ken Moody. 2004. Using trust and risk in role-based access control policies. In *SACMAT*. ACM, 156–162.

[16] Daniel Ricardo dos Santos, Silvio Ranise, Luca Compagna, and Serena Elisa Ponta. 2017. Automatically finding execution scenarios to deploy security-sensitive workflows. *J. Journal of Computer Security* 54, 3 (2017), 255–282.

[17] R.G. Downey and M.R. Fellows. 2013. *Fundamentals of Parameterized Complexity*. Springer.

[18] Philip W. L. Fong. 2019. Results in Workflow Resiliency: Complexity, New Formulation, and ASP Encoding. In *CODASPY*. ACM, 185–196.

[19] Russell Impagliazzo and Ramamohan Paturi. 1999. Complexity of k-SAT. In *Computational Complexity Conference*. IEEE Computer Society, 237–240.

[20] Stasys Jukna. 2001. *Extremal Combinatorics - With Applications in Computer Science*. Springer, Berlin.

[21] Daniel Karapetyan and Gregory Gutin. 2021. Solving the Workflow Satisfiability Problem using General Purpose Solvers. *arXiv* 2105.03273 (2021).

[22] Daniel Karapetyan, Andrew J. Parkes, Gregory Z. Gutin, and Andrei Gagarin. 2019. Pattern-Based Approach to the Workflow Satisfiability Problem with User-Independent Constraints. *J. Artif. Intell. Res.* 66 (2019), 85–122. https://doi.org/10.1613/jair.1.11339

[23] Harold W Kuhn. 1956. Variants of the Hungarian method for assignment problems. *Naval research logistics quarterly* 3, 4 (1956), 253–258.

[24] N. Li, Q. Wang, and M. V. Tripunitara. 2009. Resiliency Policies in Access Control. *ACM Trans. Inf. Syst. Secur.* 12, 4 (2009).

[25] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. 2011. Lower bounds based on the Exponential Time Hypothesis. *Bull. EATCS* 105 (2011), 41–72.

[26] John C. Mace, Charles Morisset, and Aad P. A. van Moorsel. 2014. Quantitative Workflow Resiliency. In *ESORICS (1) (Lecture Notes in Computer Science, Vol. 8712)*. Springer, 344–361.

[27] Srdjan Marinovic, Naranker Dulay, and Morris Sloman. 2014. Rumpole: An Introspective Break-Glass Access Control Language. *ACM Trans. Inf. Syst. Secur.* 17, 1 (2014), 2:1–2:32.

[28] Helmut Petritsch. 2014. *Break-Glass - Handling Exceptional Situations in Access Control*. Springer.

[29] Thomas J. Schaefer. 1978. The Complexity of Satisfiability Problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing (STOC '78)*. Association for Computing Machinery, New York, NY, USA, 216–226. https://doi.org/10.1145/800133.804350

[30] Q. Wang and N. Li. 2010. Satisfiability and Resiliency in Workflow Authorization Systems. *ACM Trans. Inf. Syst. Secur.* 13, 4 (2010), 40.

[31] M. Zavatteri and L. Vigano. 2019. Last man standing: Static, decremental and dynamic resiliency via controller synthesis. *Journal of Computer Security* 27, 3 (2019), 343–373.