Computing (2020) 102:1171–1185
https://doi.org/10.1007/s00607-019-00783-8

On privacy-aware eScience workflows

Khalid Belhajjame[1] · Noura Faci[2] · Zakaria Maamar[3] · Vanilson Burégio[4] · Edvan Soares[4] · Mahmoud Barhamgi[2]

Received: 15 May 2019 / Accepted: 21 December 2019 / Published online: 11 January 2020

Abstract
Computing-intensive experiments in modern sciences have become increasingly data-driven illustrating perfectly the Big-Data era. These experiments are usually specified and enacted in the form of workflows that would need to manage (i.e., read, write, store, and retrieve) highly-sensitive data like persons' medical records. We assume for this work that the operations that constitute a workflow are 1-to-1 operations, in the sense that for each input data record they produce a single data record. While there is an active research body on how to protect sensitive data by, for instance, anonymizing datasets, there is a limited number of approaches that would assist scientists with identifying the datasets, generated by the workflows, that need to be anonymized along with setting the anonymization degree that must be met. We present in this paper a solution privacy requirements of datasets used and generated by a workflow execution. We also present a technique for anonymizing workflow data given an anonymity degree.

Keywords Privacy · e-Science · Workflow

Mathematics Subject Classification 97R50

1 Introduction

To promote reproducibility of research, funding agencies (e.g., NSF)[1] and international initiatives such as RDA alliance,[2] started recently encouraging scientists to publish, as well as the scholarly article describing the findings of their research, the data and experiments that support the findings reported on. In the case of experiments designed and enacted using workflow systems, this essentially means publishing the

✉ Khalid Belhajjame
khalid.belhajjame@dauphine.fr

Extended author information available on the last page of the article

[1] https://www.nsf.gov/pubs/2019/nsf19022/nsf19022.pdf.

[2] https://www.rd-alliance.org.

specification of the Data Analysis Workflow (DWf) that incarnates the experiment conducted by the scientist, as well as the data used and generated by the executions of the DWf as a whole and by the steps (operations) that compose the DWf [5]. The availability of the data used and generated by the DWf executions is useful in checking the reproducibility of the DWf's results since it allows scientists to explore and verify that the DWf's results can be replicated, and are, therefore, trustworthy.

In certain modern sciences, such as social science and medical science, DWfs incarnate experiments that manipulate sensitive data about individuals. Because of this, scientists do not publish the data used and generated by the DWf executions. This is a hurdle towards enabling reproducibility since third party scientists are unable to explore at all the data used and generated by the DWf executions.

To address this problem, we investigate in this paper the problem of sanitizing the data used and generated by the DWfs that manipulate sensitive datasets, and we do so using k-anonymization. The obtained anonymized data may not allow the scientist to repeat the experiment, i.e., re-execute the DWf using the initial (non-anonymized) datasets and verify that the execution delivers final results that are identical (or at least similar) to those reported on. That said, the obtained anonymized data allow third-party scientists to explore, and ultimately verify the claims made by the authors in the scholarly article. In particular, we argue that the data used and generated by DWfs, albeit anonymized, allow the scientist to "go back and see what happened" in the data analysis and gain a better understanding of how and why the results were produced.

Our objective is not to invent a new technique for anonymizing the data used and generated by DWfs. Instead, we show how an existing technique, specifically k-anonymization, can be leveraged for this purpose. Specifically, the novelty of our solution consists of providing assistance to those who have to share datasets that have been used and generated by DWfs with peers without being sure how these datasets will be used. In doing so, we address the following two issues:

1. Not all the inputs and outputs of the operations that compose a DWf carry sensitive information about individuals at run time. The user has to manually identify the parameters that carry sensitive information. Moreover, s/he needs to specify the anonymity degree (k) that needs to be applied to the data instances of those parameters. This task can be tedious, since the user may need to examine all parameters for each operation in the DWf.
2. K-anonymization techniques expect as input a single relational table, whereas the data used and generated by the executions of a DWf is scattered into multiple tables, each representing the data instances of a given operation input or output.

To address the above two issues, we make the following novel contributions. Firstly, we propose and develop a technique for automatically identifying the parameters of the DWf that may be sensitive. The proposed technique also allows for the automatic identification of the anonymity degree that needs to be enforced. In particular, the technique identifies sensitivity and anonymity-degree by leveraging workflow parameter dependencies derived from the dataflow. Secondly, we developed a technique for merging the data generated by the DWf's parameters in a principled manner into what we call a global table, prior to using a state of the art k-anonymization technique. Note that this article is an extension of a previously published paper [3], in which we

presented a technique for specifying privacy requirements. In this paper, we go on to show how the data produced by the DWf executions can actually be anonymized.

It is worth underlining that we assume that the operations that constitute a workflow are 1-to-1 operations, in the sense that for each input data record they produce a single data record. Collection-based operations that take and/or produce collections of records are outside the scope of this paper and are the subject of ongoing work. The paper is organized as follows. Section 2 presents a DWf from the health-care domain as a running example. Section 3 outlines our approach. Section 4 presents a new technique for automatically detecting sensitive workflow parameters, and for identifying the anonymity degree that should be enforced. Section 5 details this technique for anonymizing the data used and generated by the DWf's executions given an anonymity degree. Section 6 reports on validation exercises that we conducted. Section 7 presents a literature review and some concluding remarks.

2 Running scenario

Figure 1 illustrates a DWf that consists of five operations ($op_{i=1,5}$) connected through data dependencies. Input/Output parameters are omitted for the sake of readability. This workflow's operations are as follows. op_1 queries a dataset in order to obtain nutrition data about patients, and op_2 retrieves oncology data about patients in terms of type of cancer and age (see Table 1). op_3 combines the data output by op_1 and op_2. Specifically, it performs a natural join on nutrition and oncology information. The outcome is presented in Table 2. op_4 establishes correlations according to the needs of doctors. Specifically, based on a patient's nutritional details and health conditions, op_4 generates an indicator, which specifies if the nutritional habits of the patient need to be revisited.

3 Approach to anonymizing workflow data

Figure 2 illustrates our 2-step approach to anonymize workflow data. Given a workflow specification, the first step, namely Privacy requirement identification, consists of assisting the workflow user to specify privacy requirements that need to be enforced when anonymizing the data obtained as a result of the workflow execution. Given a repository containing the data used and generated by the executions of the workflow (and its constituent operations), the second step, namely Data anonymization, consists of helping the workflow user anonymize such data in a principled manner before it is shared or published.

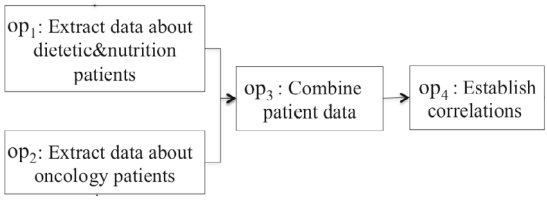

Fig. 1 Illustrative data-analysis workflow

Table 1 Nutritional details about patients (the table on the left), and oncology details about patients (the table on the right)

Patient	ID	Veg (g)	Dairy (cl)	Meat (g)
John	1	80	33	150
Ahmed	2	100	20	200
Ian	3	100	50	300
Suzanne	4	50	50	400
Yassmine	5	300	0	0
Xin	6	250	0	0

Patient	ID	Type of cancer	Age
John	1	Melanoma	25
Ahmed	2	Lung cancer	28
Ian	3	lymphoma	35
Suzanne	4	Breast cancer	40
Yassmine	5	Cervical cancer	65
Xin	6	Ovarian cancer	70

Table 2 Combined nutritional and oncology details about patients

Patient	ID	Age	Cancer	Veg (g)	Dairy (cl)	Meat (g)
John	1	25	Melanoma	80	33	150
Ahmed	2	28	Lung cancer	100	20	200
Ian	3	35	Lymphoma	100	50	300
Suzanne	4	40	Breast cancer	50	50	400
Yassmine	5	65	Cervical cancer	300	0	0
Xin	6	70	Ovarian cancer	250	0	0

Different techniques can be used for data anonymization (e.g. [7,8,12,16,17]. Differential privacy [8] is "perhaps" the most sophisticated technique in terms of privacy guarantees. That said, it is not suitable for our purpose since it protects individual privacy in the context of statistical queries, only. In our case, we are interested in providing users with the means to explore data produced following the executions of a workflow, as opposed to generating some statistics, which is what differential privacy is mainly targeted for. Because of differential privacy's limitation, we adopt k-anonymity, which is still widely used for relational data anonymization. To illustrate how k-anonymity operates, let us consider a dataset (d) of records referring each to an individual, e.g., age, address, and gender that could be used to reveal his identity. Such attributes are known as quasi-identifiers. (d) is k-anonymized, where (k) is an integer, if each quasi-identifier tuple occurs in at least (k) records in (d). For example, Table 3 illustrates data obtained by 2-anonymizing the data generated by op_2. Each tuple occurs at least twice in the dataset. Therefore, each patient contained in the anonymized version of (d) cannot be distinguished from at least 2 individuals. In the remainder of the paper, we use the term *anonymity degree* to refer to (k).

4 Privacy-preserving data analysis workflows

We first, formalize our model for a DWf and then specify its inputs in terms of sensitivity and anonymity degree. Finally, we present a solution that automatically identifies the sensitivity and anonymity degree of the DWf's remaining parameters.

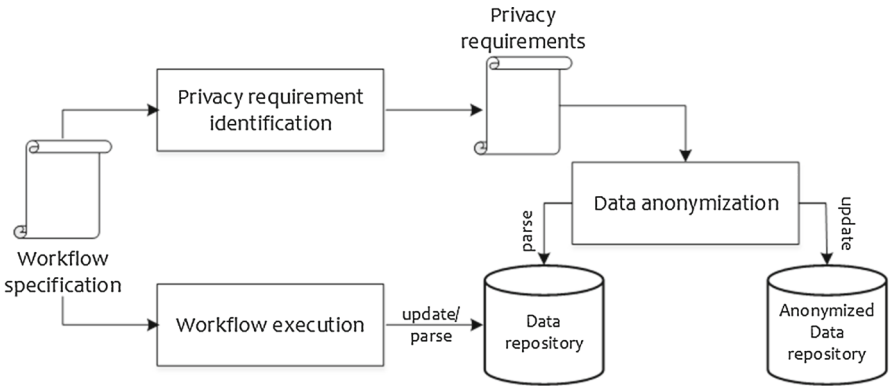

Fig. 2 Architecture

Table 3 Anonymized oncology data of patients with $\mathtt{k} = 2$

Patient	ID	Type of cancer	Age
*	*	Melanoma	$20 \leq$ age ≤ 30
*	*	Lung cancer	$20 \leq$ age ≤ 30
*	*	lymphoma	$30 <$ age ≤ 40
*	*	Breast cancer	$30 <$ age ≤ 40
*	*	Cervical cancer	$60 \leq$ age ≤ 70
*	*	Ovarian cancer	$60 \leq$ age ≤ 70

4.1 Workflow model definition

Workflow model A $\mathtt{DWf}$ is a tuple $\langle \mathtt{DWf}_{id}, \mathtt{OP}, \mathtt{DL} \rangle$ where $\mathtt{DWf}_{id}$ is a unique identifier, $\mathtt{OP}$ is a set of data manipulation operations ($\mathtt{op}_i$), and $\mathtt{DL}$ is the set of data dependencies between these operations.

Operation $\mathtt{op}_i$ is defined by $\langle \mathtt{name}, \mathtt{in}, \mathtt{out} \rangle$ where $\mathtt{name}$ is self-descriptive, and $\mathtt{in}$ and $\mathtt{out}$ are self-descriptive, too. As some output parameters could be other operations' inputs, a parameter has a unique name ($\mathtt{p}_{\mathtt{name}}$).

Let $\mathtt{IN} = \cup_{\mathtt{op} \in \mathtt{OP}}(\mathtt{op.in})$ and $\mathtt{OUT} = \cup_{\mathtt{op} \in \mathtt{OP}}(\mathtt{op.out})$ be the sets of all operations' inputs and outputs in a $\mathtt{DWf}$, respectively. The set of data dependencies connecting the workflow operations must then satisfy $\mathtt{DL} \subseteq (\mathtt{OP} \times \mathtt{OUT}) \times (\mathtt{OP} \times \mathtt{IN})$. A data dependency relating $\mathtt{op}_1$'s output $\langle \mathtt{o}, \mathtt{op}_1 \rangle$ to $\mathtt{op}_2$'s input $\langle \mathtt{i}, \mathtt{op}_2 \rangle$ is denoted by the pair $\langle \langle \mathtt{o}, \mathtt{op}_1 \rangle, \langle \mathtt{i}, \mathtt{op}_2 \rangle \rangle$. We use $\mathtt{IN}_{\mathtt{DWf}}$ and $\mathtt{OUT}_{\mathtt{DWf}}$ to denote $\mathtt{DWf}$'s inputs and outputs, respectively. In this work, we consider acyclic workflows that are free of loops.

Parameter sensitivity To specify that a ($\mathtt{DWf}$)'s given input or output parameter holds sensitive data, the workflow designer annotates these data for that purpose. In the running example, the designer annotates the two initial parameters (i.e., collections of records about patients along with their nutrition and cancer histories) of the workflow as sensitive in their respective instances of.

Given those annotations, we assume the existence of a Boolean function: $\texttt{isSensitive}(\langle \texttt{op}, \texttt{p} \rangle)$, that is true if the annotation associated with $\langle \texttt{op}, \texttt{p} \rangle$ states that it is sensitive; and, false, otherwise.

Regarding the remaining parameters in the workflow, we assist the workflow designer in the annotation task by automatically identifying the parameters that may be sensitive by leveraging parameters' dependencies, as we will see in the next section.

Parameter anonymity degree The execution of a DWf refers to a DWf instance denoted by (insWf). The anonymity degree of a DWf's parameter ($\langle \texttt{p}, \texttt{op} \rangle$) is defined with respect to a given DWf instance (insWf). Indeed, multiple instances of the same DWf may have as input datasets different anonymity degree requirements. For example, the owner of an input dataset used for a given workflow instance ($\texttt{insWf}_1$) may impose a more stringent anonymity degree than another owner of an input dataset used for a different workflow instance ($\texttt{insWf}_2$). As a result, the same workflow parameter may have different anonymity degrees depending on the workflow instance in question. Due to this difference in requirement, we use the following function to specify the anonymity degree of a given parameter $\langle \texttt{p}, \texttt{op} \rangle$ with respect to a workflow instance insWf: $\texttt{anonymity}(\langle \texttt{p}, \texttt{op} \rangle, \texttt{insWf})$.

For example, $\texttt{anonymity}(\langle \texttt{p}, \texttt{op}_1 \rangle, \texttt{insWf}_1) = 3$ specifies that $\langle \texttt{p}, \texttt{op}_1 \rangle$ has an anonymity degree of 3 within the workflow instance $\texttt{insWf}_1$. Consider that the dataset d is bound to $\langle \texttt{p}, \texttt{op}_1 \rangle$ in $\texttt{insWf}_1$. Given that $\texttt{anonymity}(\langle \texttt{i}, \texttt{op}_1 \rangle, \texttt{insWf}_1) = 3$, (d) must be anonymized before its publication. Specifically, each record (individual) in the anonymized (d) must not be distinguished from at least (2) other individuals [16].

4.2 Detecting sensitive parameters and identifying their anonymity degrees

Manual identification of the DWf's parameters that are sensitive and setting their anonymity degrees can be tedious. This becomes a serious concern when the DWf includes a large number of operations. To address this issue, we propose an approach that detect operation parameters that may be sensitive, and the anonymity degree that needs to be considered. In doing so, we leverage the dependencies between the operation parameters that are extracted from the dataflow of the DWf to identify if a given parameter is sensitive by checking if it (transitively) depends on an input parameter of the DWf that is sensitive. Similarly, we exploit parameter dependencies to identify the value of anonymity degree that if adopted will allow honoring the anonymity degrees associated with the parameters of all the operations in the DWf .

Parameter dependencies Dependencies between a workflow DWf's parameters is a key element to our approach. A parameter $\langle \texttt{op}, \texttt{p} \rangle$ depends on a parameter $\langle \texttt{op}', \texttt{p}' \rangle$ in a workflow DWf, if during the execution of DWf the data bound to the parameter $\langle \texttt{op}', \texttt{p}' \rangle$ contribute to or influence the data bound to the parameter $\langle \texttt{op}', \texttt{p}' \rangle$.[3]

[3] The notion of contribution and influence are in line with the derivation and influence relationship defined by the W3C PROV recommendation [14].

Parameter dependencies can be specified by examining the workflow specification $\texttt{DWf}$.[4] Given a workflow ($\texttt{DWf}$), the dependencies between its parameters are extracted as follows:

– Consider for instance that $\langle \texttt{i}, \texttt{op} \rangle$ and $\langle \texttt{o}, \texttt{op} \rangle$ are input and output. We say that $\langle \texttt{o}, \texttt{op} \rangle$ depends on $\langle \texttt{i}, \texttt{op} \rangle$, which we write:

$$\texttt{dependsOn}(\langle \texttt{o}, \texttt{op} \rangle, \langle \texttt{i}, \texttt{op} \rangle)$$

– If the workfow ($\texttt{DWf}$) contains a data link connecting $\langle \texttt{op}, \texttt{o} \rangle$ to $\langle \texttt{op}, \texttt{i} \rangle$, then we say that $\langle \texttt{op}, \texttt{i} \rangle$ depends on $\langle \texttt{op}, \texttt{o} \rangle$, i.e., $\texttt{dependsOn}(\langle \texttt{o}, \texttt{op} \rangle, \langle \texttt{i}, \texttt{op}' \rangle)$.

We also transitively derive dependencies between the operation parameters of a workflow based on the following rules:

$$R_1\ \texttt{dependsOn}^*(\langle \texttt{p}, \texttt{op} \rangle, \langle \texttt{p}', \texttt{op}' \rangle) :- \texttt{dependsOn}(\langle \texttt{p}, \texttt{op} \rangle, \langle \texttt{p}', \texttt{op}' \rangle)$$
$$R_2\ \texttt{dependsOn}^*(\langle \texttt{p}, \texttt{op} \rangle, \langle \texttt{p}', \texttt{op}' \rangle) :- \texttt{dependsOn}^*(\langle \texttt{p}, \texttt{op} \rangle, \langle \texttt{p}'', \texttt{op}'' \rangle), \texttt{dependsOn}^*(\langle \texttt{p}'', \texttt{op}'' \rangle, \langle \texttt{p}', \texttt{op}' \rangle)$$

Applying the above rules to our example workflow, we conclude for instance, $\texttt{dependsOn}^*(\langle \texttt{o}, \texttt{op}_3 \rangle, \langle \texttt{i}, \texttt{op}_2 \rangle)$, where $\texttt{i}$ and $\texttt{o}$ are parameter names.

Detecting sensitive parameters We use parameter dependencies to assist the workflow designer identify the intermediate and final parameters that may be sensitive. Specifically, a parameter $\langle \texttt{p}', \texttt{op}' \rangle$ that is not an input to the workflow, i.e., $\langle \texttt{p}', \texttt{op}' \rangle \notin IN_{DWf}$, may be sensitive if it depends on a workflow input that is known to be sensitive, i.e.,

$$\exists \langle \texttt{i}, \texttt{op} \rangle \in \texttt{IN}_{\texttt{DWf}}\ \texttt{s.t.}\ \texttt{sensitive}(\texttt{i}, \texttt{op}) \wedge \texttt{dependsOn}^*(\langle \texttt{p}', \texttt{op}' \rangle, \langle \texttt{i}, \texttt{op} \rangle)$$

Note that we say that $\langle \texttt{p}', \texttt{op}' \rangle$ *may be* sensitive. This is because an operation that consumes sensitive datasets may produce non-sensitive datasets. For example, $\texttt{op}_5$ in Fig. 1 generates non-sensitive data although its outputs are sensitive inputs to a workflow. The output of such an operation is a report that is free from information about individual patients.

Identifying anonymity degree In addition to assisting the designer with identifying sensitive intermediate and final output parameters, we also identify the anonymity degree that should be applied to dataset instances of those sensitive parameters. To illustrate this, consider that $\langle \texttt{p}', \texttt{op}' \rangle$ is a sensitive intermediate or final output parameter. The anonymity degree of such a parameter given a workflow execution $\texttt{insWf}$ can be defined as the maximum degree of the sensitive datasets that are used as input to the workflow and that contribute to the dataset instances of $\langle \texttt{p}', \texttt{op}' \rangle$. Taking the maximum anonymity degree of the contributing inputs ensures that the anonymity degrees imposed on such inputs is honored by the dependent parameter in question. That is:

[4] Parameter dependencies correspond to what is referred to in the scientific workflow community by prospective provenance. This is because such dependencies can be extracted from the workflow specification as opposed to other kinds of information, e.g., execution log, which can only be obtained retrospectively once the workflow execution terminates.

$$\texttt{anonymity}(\langle \texttt{p}', \texttt{op}' \rangle, \texttt{insWf}) = \max(\{\texttt{anonymity}(\langle \texttt{i}, \texttt{op} \rangle, \texttt{insWf}) \text{ s.t. } \texttt{sensitive}(\langle \texttt{i}, \texttt{op} \rangle) \wedge \texttt{dependsOn}^*(\langle \texttt{p}', \texttt{op}' \rangle, \langle \texttt{i}, \texttt{op} \rangle)\})$$

5 Anonymizing datasets

We have, thus far, presented a solution for detecting parameters that may be sensitive, and for identifying the anonymity degree that should be used when anonymizing the workflow datasets. In doing so, we did not actually specify how we go about anonymizing the datasets used and generated by the executions of a given DWf.

The anonymization technique that we propose is a three-step process. In the first step, we construct a schema of a global table (relation). Such a table is global in the sense that its schema union all the input and output parameters of the workflow and the steps thereof. Secondly, we populate this table using the datasets used and generated by the workflow in question over its executions. Finally, we anonymize the obtained populated global table using the anonymity degree identified using the detection technique presented in Sect. 4.2. We detail in what follows each of the above steps.

Creating the schema of the global table The objective in this step is to create a table (relation) whose schema contains the attributes of all the parameters of the operations that compose the workflow. In doing so, we identify and remove duplicates attributes. To illustrate this, consider the example of the workflow illustrated in Fig. 1. Patient ID is an attribute that characterizes the input parameters and the output parameter of CombinePatientData operation. When constructing the schema of the global table, such an attribute will appear only once. It is worth noting here that two attributes that characterize different parameters and have the same name may have different semantics, and *vice-versa*. For instance, an attribute named ID may refer to a patient or to a product. Conversely, two attributes named P_ID and Pat_ID may both represent patient identifiers.

Entity resolution techniques (see [9] for a survey) can be used for detecting parameter attributes that have the same semantics. While effective, we use in our work a technique that is simple, yet more effective and efficient given the domain knowledge that we have about the attributes in the table. Indeed, we know that two attributes are semantically the same if they contain the same data values. This technique is efficient. Given n invocations of an operation op, which has an input parameter with N_{in} attributes and an output parameter with N_{out}, the number of comparisons required to detect identical attributes is $n \cdot N_{in} \cdot N_{out}$. In other words, the number of comparisons to be made to identify identical attributes is linear with respect to the number of op invocations.

Populating the global table Populating the global table refers to the operation by which the data records (i.e., instances of the input and output parameters of the operations composing the workflow) are put together (or combined) within the table. In our work, we rely on lineage information to combine the data records of different operations within the global table. Let us consider two attributes in the schema of the global table

att_i and att_j, and att_i's and att_j's instances v_i and v_j, respectively. v_i and v_j belong to the same tuple in the global table if one of the following conditions holds:

- att_i and att_j characterize the input or output of an operation op, and v_i and v_j are associated with the same invocation of op.
- att_i and att_j characterize parameters of different operations. However v_i is derived from v_j lineage-wise, or *vise versa*.

Anonymizing the global table Once anonymity degree is computed and the global table has been populated, our technique uses an anonymization algorithm proposed in the literature like Mondarian [13] for its anonymization before publishing it. Without loss of generality, in our work, we use the ARX Data Anonymization tool.[5]

6 Validation

We report, in this section, on the validation of our solution. We start in Sect. 6.1 by assessing the performance of our solution for the identification of sensitive parameter and anonymity degree. We then go on, in Sect. 6.2, to present a use case that shows how our solution can be utilized for effectively anonymizing the data of DWfs' executions.

6.1 Anonymity degree and sensitive parameter identification

Our objective is to assess the overhead of the operation required for identifying sensitive parameters for computing the anonymity degree to be used for the anonymization of data obtained by DWf's executions. In particular, we computed parameters such as workflow loading times, time required for identifying parameter dependencies and sensitive parameters, and for computing anonymity degree. Number of operations, sensitive inputs, and anonymity degrees highlight the differences between these workflows.

For our experiment, we used 20 different CWL workflows[6] (500 executions per workflow). We used these workflows because they are publicly available, and also because they contain workflows with different numbers of operations and data links. The smallest workflow contains a single operation and no data link, and the largest one contains 29 operations and 50 data links. In average, a workflow in the set we used contains 6 operations and 8 data links.

For each workflow, we computed the minimum, maximum, and average overhead due to workflow loading, parameter dependency extraction, sensitive parameter identification, and anonymity degree computation, across the 10 K executions. Figure 3 illustrates the time required for extracting parameter dependencies from the workflow specification. Notice that the required minimum and average time can be hardly seen on the chart; in fact, the extraction of dependencies is instantaneous in most cases. For the required maximum time, it is less than 0.2 ms for most workflows. However, 3 outliers have been identified, workflows 2, 13, and 20, that take almost 15 ms in the

[5] arx.deidentifier.org.

[6] view.commonwl.org/workflows.

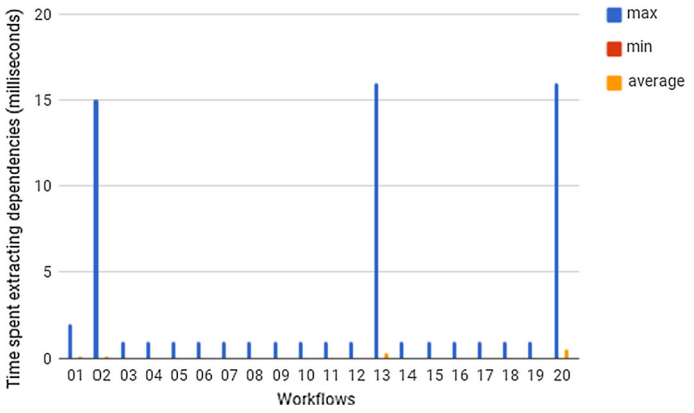

Fig. 3 Overhead due to parameter dependency extraction

worst case. This can be explained by the fact that dependency extraction is influenced by the number of input and output parameters the workflow has. The examination of workflows 2, 13, and 20 revealed that they have a larger number of outputs compared to the rest of workflows.

Regarding the overhead due to workflow loading, sensitive parameter detection and anonymity degree calculation, it is almost instantaneous for all workflows, and therefore there was no need to show the charts for them (also due to limited space). In summary, the experimental results are encouraging and show that the overhead due to the solution can barely be noticed.

6.2 Use case for anonymizing workflow data

To showcase how the technique for anonymizing workflow datasets described in the previous subsections work, we will use in this section the example workflow illustrated in Fig. 1. We have used for this use case synthetic data that we generated for this purpose. Using data about real individuals with information about their health conditions cannot be published as-is. On the other hand, synthetic data that we generated for validation purposes are publicly available online.[7] We assume that the anonymity degree to apply to the data produced by the workflow is 2. In what follows, we illustrate how each step of workflow data anonymization proceeds.

Step 1: Creation of the schema of the global table The specification of the workflow illustrated in Fig. 1 is composed of four operations. The attributes that form the input and output of each operation are illustrated below:

- $\mathtt{op}_1 : \langle \mathtt{ID} \rangle \rightarrow \langle \mathtt{Patient}, \mathtt{ID}, \mathtt{Veg}, \mathtt{Dairy}, \mathtt{Meat} \rangle$
- $\mathtt{op}_2 : \langle \mathtt{ID} \rangle \rightarrow \langle \mathtt{Patient}, \mathtt{ID}, \mathtt{TypeofCancer}, \mathtt{Age} \rangle$
- $\mathtt{op}_3 : \langle \langle \mathtt{Patient}, \mathtt{ID}, \mathtt{Veg}, \mathtt{Dairy}, \mathtt{Meat} \rangle, \langle \mathtt{Patient}, \mathtt{ID}, \mathtt{Age}, \mathtt{TypeofCancer} \rangle \rangle \rightarrow \langle \mathtt{Patient}, \mathtt{ID}, \mathtt{Age}, \mathtt{TypeofCancer}, \mathtt{Veg}, \mathtt{Dairy}, \mathtt{Meat} \rangle$

[7] https://www.lamsade.dauphine.fr/~belhajjame/data.

– $\mathtt{op}_4 : \langle \mathtt{Patient}, \mathtt{ID}, \mathtt{Age}, \mathtt{TypeofCancer}, \mathtt{Veg}, \mathtt{Dairy}, \mathtt{Meat} \rangle \rightarrow \langle \mathtt{Indicator} \rangle$

Notice that the attributes Patient and ID appear multiple times in the operation signatures, with the same semantics. Moreover, they hold the same information for a given patient all along with the execution of the workflow, i.e., ID (resp. Patient) has the same value across a workflow execution. Because of this, the global table will contain only one occurrence of each of these attributes. Notice also that the values of the attributes used and generated by the operation $\mathtt{op}_3$ have the same semantics. This is specifically the case for the attributes ID, Veg, Dairy, Meat, TypeofCancer and Age. These attributes will, therefore, appear only once in the schema of the global table. Based on the above discussion, we derive the following schema for the global table, which we refer to as GT.

GT(Patient, ID, Age, TypeofCancer, Veg, Dairy, Meat, Indicator)

Step 2: Populating the global table Based on workflow lineage information, we populate the global table. Indeed, major workflow systems are instrumented to capture lineage information specifying which operation input values were used to generate given output values. Given this information in our example workflow, we were able to generate the global table illustrated in Table 4.

Step 3: Anonymizing the global table In this step, we applied an existing k-anonymization algorithm (namely the ARX tool)[8] to our global table using the anonymity degree $\mathtt{k} = 2$. The result is illustrated in Table 4. Notice that the name of the patient and its ID have been removed. Note also that the attributes TypeofCancer and Indicator were annotated as sensitive. Their values have not been generalized. Instead, the values of the remaining attributes, which were annotated as quasi-identifying attributes have been generalized to meet the anonymity degree of 2. As a result, one cannot distinguish between at least two patients based on the values of the quasi-identifying attributes.

Note that the solution we present caters for the cases where the operations do some munging and merging of the input data. In such cases, the attributes of the data output by such an operation can also be quasi-identifiers and, therefore, need to be generalized. Take, for instance, the operation "Establish Correlations" in the example workflow. Such an operation produces an indicator, which specifies if the nutritional habits of the patient need to be revisited. In the example, such an output take traffic-light values to indicate if the nutrition habits of the patient in question need to be drastically changed (red), slightly changed (orange). Otherwise, it takes the value green indicating that the nutritional habits are healthy. Such an indicator can take more specific values to indicate, for example, the intake in veg, dairy and meat needs to be lowered or increased for a given patient.

The above example, albeit small, illustrated how our approach operates. We chose a small example to make it easier for the reader to grasp the solution. That said, our approach can scale to workflows with large datasets. Indeed, as pointed out earlier both the construction of the global table and its population are performed in a time that is linear to the number of operations that compose the workflow.

[8] https://arx.deidentifier.org.

Table 4 The table on the top represents the populated global table, and the table in the bottom represents the anonymized global table with $k = 2$

Patient	ID	Age	Cancer	Veg (g)	Dairy (cl)	Meat (g)	Indicator
John	1	25	Melanoma	80	33	150	Orange
Ahmed	2	28	Lung cancer	100	20	200	Orange
Ian	3	35	Lymphoma	100	50	300	Red
Suzanne	4	40	Breast cancer	50	50	400	Red
Yassmine	5	65	Cervical cancer	300	0	0	Green
Xin	6	70	Ovarian cancer	250	0	0	Green

Patient	ID	Age	Type of cancer	Veg	Dairy	Meat	Indicator
*	*	$20 \leq \text{age} \leq 30$	Melanoma	$80\,\text{g} \leq \text{veg} \leq 100\,\text{g}$	$20\,\text{cl} \leq \text{dairy} < 40\,\text{cl}$	$100\,\text{g} \leq \text{meat} \leq 200\,\text{g}$	Orange
*	*	$20 \leq \text{age} \leq 30$	Lung cancer	$80\,\text{g} \leq \text{veg} \leq 100\,\text{g}$	$20\,\text{cl} \leq \text{dairy} < 40\,\text{cl}$	$100\,\text{g} \leq \text{meat} \leq 200\,\text{g}$	Orange
*	*	$30 < \text{age} \leq 40$	Lymphoma	$0\,\text{g} \leq \text{veg} \leq 50\,\text{g}$	$40\,\text{cl} < \text{dairy} \leq 50\,\text{cl}$	$200\,\text{g} < \text{meat} \leq 400\,\text{g}$	Red
*	*	$30 < \text{age} \leq 40$	Breast cancer	$0\,\text{g} \leq \text{veg} \leq 50\,\text{g}$	$40\,\text{cl} < \text{dairy} \leq 50\,\text{cl}$	$200\,\text{g} < \text{meat} \leq 400\,\text{g}$	Red
*	*	$60 \leq \text{age} \leq 70$	Cervical cancer	$200\,\text{g} \leq \text{veg} \leq 300\,\text{g}$	$0\,\text{cl} \leq \text{dairy} < 20\,\text{cl}$	$0\,\text{g} < \text{meat} \leq 50\,\text{g}$	Green
*	*	$60 \leq \text{age} \leq 70$	Ovarian cancer	$200\,\text{g} \leq \text{veg} \leq 300\,\text{g}$	$0\,\text{cl} \leq \text{dairy} < 20\,\text{cl}$	$0\,\text{g} < \text{meat} \leq 50\,\text{g}$	Green

7 Related work and concluding remarks

7.1 Privacy concerns in scientific workflows

We present in this section proposals that tackle privacy concerns in scientific workflow and conclude the section by discussing how our work advances the state of the art.

Gil et al. [11] address the issue of data privacy in the context of DWfs. To this end, they propose an ontology that preserves this privacy along with enforcing access control over data with respect to a given set of access permissions. The ontology specifies eligible privacy-preserving policies (e.g., generalization and anonymization) per DWf′s input/output parameter. To support privacy policy enforcement in DWfs, a framework was developed to represent policies as a set of elements that include applicable context, data usage requirement, privacy protection requirement, and corrective actions if the policy is violated.

Chebbi and Tata [4] propose a workflow reduction-based abstraction approach for workflow advertisement purposes. The approach reduces a workflow inter-visibility using 13 rules that depend on dependencies between operations in the workflows along with the operation types (i.e., internal vs. external).

Alhaqbani et al. [1] propose a privacy-enforcement approach for business workflows based on 4 requirements: (i) capture the *subject* (i.e., data owner)'s privacy policy during the workflow specification on top of the privacy policies defined by the workflow administrator, (ii) define data properties (i.e., hide and generalize) linked to private data so that these properties influence the workflow engine to protect data as per the *subject*'s privacy policy, (iii) allocate work while preserving privacy, i.e., assign the task referring to some manipulation of data, to the employee who has the lowest restriction level according to the *subject*'s privacy policy, and (iv) keep the subject informed about any attempt for accessing his/her data.

Barth et al. [2] present a privacy-policy violation detection approach based on execution logs of business processes. The aim is to identify a set of employees potentially responsible for privacy breach. The authors introduce two types of compliance: strong and weak. An action is strongly compliant with a privacy policy given a trace if there exists an extension of the trace that contains the action and satisfies the policy. An action is weakly compliant with a policy given a trace if the trace augmented with the action satisfies the present requirements of the privacy policy.

Davidson et al. [6] discuss privacy-preserving management of provenance-aware workflow systems. The authors first formalize the privacy concerns: (i) *data privacy* that requires outputs of the workflow's operations should not reveal to users without an access privilege, (ii) *module privacy* that requires the functionality of an operation is not revealed, and (iii) *structural privacy* that refers to hiding the data flow's structure in the given execution.

The above proposals can be classified into two categories. Those that preserve the privacy of tasks (operations) of workflows. This is exemplified in the works by Barth et al. [2] and Davidson et al. [6]. And those that preserve the privacy of data that workflows manipulate at run-time. This is exemplified with the works of Gil et al. [11] and Alhaqbani et al. [1]. Contrarily, the work of Sharif et al. [15] addresses the privacy

of both task and data. In the context of our work, we are concerned with the privacy of workflow data and hence, is in line with the second category of proposals. However, achieving this privacy requires that the workflow designer manually identifies sensitive workflow parameters and sets the degree to which the datasets bound to those parameters need to be anonymized. We have taken care of both aspects of our work.

7.2 Data anonymization

Data anonymization is the operation by which datasets are processed so that the persons whom the data describe remain anonymous. There are five main anonymization operations [10], namely generalization, suppression, anatomization, permutation, and perturbation. Generalization replaces attribute values with less specific but semantically consistent values [17]. Suppression, on the other hand, masks completely the values of given attributes [7]. This is the technique used, for example, by Davidson et al. [6]. Anatomization is used to dissociate the correlation that may be observed between the quasi-identifier attributes and sensitive attributes by separating them into different datasets. The objective of permutation is the same as anatomization but do so by dividing a number of data records into groups and mixing their sensitive values in every group. Perturbation modifies attribute values with new values by interchanging attribute values, adding noise or creating synthetic data. For example, the differential privacy technique anonymizes data using perturbation.

In our work, we assumed that the operations that constitute a workflow are 1-to-1 operations, in the sense that for each input data record they produce a single data record. In our ongoing work, we are investigating privacy in the context of collection-based workflows, the operations of which can consume and/or generate collections of data records.

References

1. Alhaqbani B, Adams M, Fidge CJ, ter Hofstede AHM (2013) Privacy-aware workflow management. In: Glykas M (ed) Business process management. Springer, Dortmund, pp 111–128
2. Barth A, Mitchell JC, Datta A, Sundaram S (2007) Privacy and utility in business processes. In: CSF. IEEE, pp 279–294
3. Belhajjame K, Faci N, Maamar Z, Burégio VA, Soares E, Barhamgi M (2019) Privacy-preserving data analysis workflows for escience. In: Proceedings of the CEUR workshop of the EDBT/ICDT, vol 2322. CEUR-WS.org
4. Chebbi I, Tata S (2007) Workflow abstraction for privacy preservation. In: Weske M, Hacid MS, Godart C (eds) International conference on web information systems engineering. Springer, Nancy, pp 166–177
5. Cohen-Boulakia S, Belhajjame K, Collin O, Chopard J, Froidevaux C, Gaignard A et al (2017) Scientific workflows for computational reproducibility in the life sciences: status, challenges and opportunities. Future Gener Comput Syst 75:284–298
6. Davidson SB, Khanna S, Tannen V, Roy S, Chen Y, Milo T, Stoyanovich J (2011) Enabling privacy in provenance-aware workflow systems. In: Biennial conference on innovative data systems research, pp 215–218
7. Dolby RB, Harvey G, Jenkins NP, Raviraj R (2000) Data suppression and regeneration. US Patent 6,038,231

8. Dwork C (2006) Differential privacy. In: Proceedings on 33rd international colloquium on automata, languages and programming, part II. Springer, Venice, pp 1–12
9. Elmagarmid Ahmed K, Ipeirotis Panagiotis G, Verykios Vassilios S (2007) Duplicate record detection: a survey. IEEE Trans Knowl Data Eng 19(1):1–16
10. Eyupoglu Can, Aydin Muhammed Ali, Zaim Abdul Halim, Sertbas Ahmet (2018) An efficient big data anonymization algorithm based on chaos and perturbation techniques. Entropy 20(5):373
11. Gil Y, Cheung WK, Ratnakar V, Chan K-K (2007) Privacy enforcement in data analysis workflows. In: AAAI workshop on privacy enforcement and accountability with semantics. AAAI, Busan, pp 41–48
12. Kargupta H, Datta S, Wang Q, Sivakumar K (2003) On the privacy preserving properties of random data perturbation techniques. In: ICDM. IEEE, pp 99–106
13. LeFevre K, DeWitt DJ, Ramakrishnan V (2006) Mondrian multidimensional k-anonymity. In: International conference on data engineering. IEEE, Atlanta, p 25
14. Missier P, Belhajjame K, Cheney J (2013) The W3C PROV family of specifications for modelling provenance metadata. In: EDBT/ICDT. ACM press, pp 773–776
15. Sharif S, Taheri J, Zomaya AY, Nepal S (2013) MPHC: preserving privacy for workflow execution in hybrid clouds. In: PDCAT. IEEE, pp 272–280
16. Sweeney L (2002) k-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl Based Syst 10(05):557–570
17. Wang K, Yu PS, Chakraborty S (2004) Bottom-up generalization: a data mining solution to privacy protection. In: ICDM. IEEE, pp 249–256

Affiliations

Khalid Belhajjame[1] · Noura Faci[2] · Zakaria Maamar[3] · Vanilson Burégio[4] · Edvan Soares[4] · Mahmoud Barhamgi[2]

Noura Faci
noura.faci@univ-lyon1.fr

Zakaria Maamar
zakaria.maamar@zu.ac.ae

Vanilson Burégio
vanilson.buregio@ufrpe.br

Edvan Soares
edvan.soares@ufrpe.br

Mahmoud Barhamgi
mahmoud.barhamgi@univ-lyon1.fr

1 LAMSADE, PSL, Université Paris-Dauphine, Paris, France

2 LIRIS, UDL, Université de Lyon, Lyon, France

3 Zayed University, Dubai, United Arab Emirates

4 Federal Rural University of Pernambuco, Recife, Brazil