27/03/2019

Dear Reviewer,

Thank you for your comments, which we have addressed, and which we believe improve out paper. Below we repeat your points and set out the responses to each weakness that you identified. Our response to each is in red.

Yours sincerely,

Jeremy Bryans (on behalf of the author team)

Weaknesses:

W1. The motivation scenario is quite weak. The authors provide two rationales for abstracting provenance information: complexity and selective disclosure. The objective of this paper is to abstract primarily due to selective disclosure as presented in the motivating example. However, the motivating examples provides no basis for subgraph obfuscation. Para2 on page 3 provide some reasons for eliminating node information but they are all node property information not rationale for subgraph obfuscation. Figure 2 is nearly unreadable. It is not clear what real world phenomenon this graph represents. It is not obvious why a user in Figure2 would like to remove nodes consolidate AJC and consolidateBNC, and what impact will that removal have—in the sense what information would have divulged otherwise and how it will affect. The scenario is presented in loose and vague terms with significant hand waving: Pg4, para 3, “some of it may be…”

Response:

We have extensively revised the initial motivating example:

* We have focused on the goal of achieving selective disclosure while de-emphasizing the goal of simplifying the provenance graph i.e. by constructing a (PROV-valid) view. Hopefully this gives the work a sharper focus.
* To clarify how, in practice, one could select the nodes to abstract, i.e., how to choose the set V\_{gr}, we have made a reference to our older IPAW paper [17] where a policy-based mechanism to achieve exactly this is described. The mechanism is fully implemented as part of our ProvAbs tool, which is also mentioned in [17]. We think it is not within the scope of this paper to elaborate at length on this point
* Fig. 1 redrawn and hopefully cleaner and simpler, and fully commented in the text. The scenario now continues illustrating two abstraction steps in sequence (Figs. 2,3) to provide a flavor for and anticipate the kind of results one would expect when using the grouping operator.

W2: Definitions of two crucial elements are note clear. It is not obvious what Vgr represents: does it represent “a set” of nodes to obfuscate or does the user have any knowledge that there exists a path between the nodes in Vgr and so chooses only those nodes? In other words, what if Vgr nodes have no path amongst them?

If they have no path then are they replaced by as many “blank” or “null” nodes as in the set Vgr, and if such nodes for validity only retain their original type?

If so, then why can’t this property be adopted universally instead of opting for expensive operations of path closure, extend, replace.

Response:

Vgr indeed is a set of nodes to be obfuscated. The mechanism by which these are selected by the user are not discussed in depth in this paper, however we now have added a reference to our previous paper where we describe a policy-driven mechanism for node selection (which is implemented in our provAbs tool).

However, we have also added text to point out that it is reasonable to assume that the provenance owner does have some knowledge of the graph topology, and this can potentially be used to guide the Vgr node selection.

We have also clarified that, by construction, the closure over set Vgr is a subgraph with the property that each source node is connected to at least one of the sink nodes (source and sink nodes are all in Vgr and the nodes in between are added to the closure). While in the first version of the paper we have assumed that only one such subgraph exists, we now also consider the more general case where the closure produces more than one such subgraph, such that there are no paths between these subgraphs.

To account for this possibility, it is enough to apply the replace() operator independently to each subgraph, yielding one abstract node for each.

In the extreme case, we could indeed have as many such subgraphs as there are nodes in Vgr. In this case, our abstraction algorithm would indeed result in replacing each of those with an anonymized version (or “blank” nodes, as the reviewer suggests above).

This observation also helps us address the reviewer’s last objection on this point, namely by saying that “single node anonymization” is a special case of obfuscation, which does not affect graph structure. However, we argue that our obfuscation model is more general and is indeed designed to change the topology as well, and the more expensive operators are necessary to ensure validity of the resulting abstracted graph.

Why is path closure referring to “a” path between nodes and not all paths?

If the idea is indeed to obfuscate the lineage between the selected nodes, it must include all paths and not “a” path.

Thank you for pointing this out – we have now clarified this (Def. 2)

W3: The motivation for extension operator is not clear.

We have added further clarifications on pg. 14 (sec. 4.1), ie., the operator is necessary in order to preserve type consistency vis a vis the PROV model (i.e., required domain and range of the relationships).

It is not evident what is the impact of hiding information which the user did not select, especially information that was obfuscated to maintain validity? What if the non selected obfuscated content is actually the information that must be communicated between the two parties.

We have clarified this point by making references (introduction and sec 4.4) to the “obfuscation policy” that is described in detail in [17], and to the metrics of “residual utility” of the abstracted graph, also described in the same paper. We feel that those ideas are peripheral to this paper, which is entirely focused on the underpinning abstraction mechanism.

The authors have not considered what domains enforce a validity constraint and what if it is relaxed to show an partially inconsistent graph?

The notion of partially inconsistent PROV graphs is an interesting one to explore, however the focus of this work is strictly on validity-preserving PROV graph transformations. One practical reason for this is that inconsistencies / constraint violations make the resulting provenance less interoperable. Also, although we do not elaborate on this point explicitly, we would like to ensure a closure property, namely that abstracted PROV graphs can be further abstracted, with validity guarantees.

Alternatively what if nodes are replaced by a subgraph? Thus in Figure 6 instead of replacing by a single node of type e’, what if a replacement subgraph of e’->a’ is provided. This subgraph signifies some entities connect to an activity, and the edge remains unlabeled (similar to the edge in Figure 6(d)). If replaced by a sub-graph instead of an entity or activity, it can be shown that no new nodes need to be included in the obfuscation cover to maintain validity. It can also be shown that it represents the orginal subgraph correctly because the original consists of both activities and entities.

Thank you for this insight, this looks like a valid alternative approach, which we may be considering in future work. Indeed we are aware that multiple abstraction models are possible, and we hope to have a study on alternative models ready for publication in the near future.

W4: The replacement on page 13 has little evidential basis. It is an incorrect operator not a naive one, in which the structure of a graph is destroyed to lead a new node for the purpose of reducing complexity or performing selective disclosure. That operation may be a transient state in 3, 12, 18 but is never performed. I am not sure what is the basis for considering a wrong operator and terming it as a naive operation.

We realise that the use of the term “naïve” is misleading here. The point of the example was to show that not all abstractions preserve validity, and to argue that a constraint-aware operator is required to preserve validity.

We have removed the term “naïve” (we use the term “careless” as a replacement) and replaced the text on pg. 13 to explain the point more clearly.

W5: Despite a mathematical foundation, the paper lacks complexity analysis of the operators. It has no experimental results to validate how efficient are these operators on real provenance graphs. The authors have provided no basis for why these operators should be considered.

We have now added complexity analysis for the closure and extend operators, please see sec. 4.3, to show a quadratic complexity in the worst case but also noting that it will be greatly reduced in practical cases.

We also clarify that, as the purpose of the paper is to provide a foundation for validity-preserving transformations of PROV graphs, we do not offer an experimental evaluation of algorithmic complexity, as we consider it beyond our current scope (also, we feel the paper is already of sufficient length)

W6: Definition 6 on homogenous grouping is not re-constructed based on outcut and incut definitions considered earlier. Again incut and outcut are not highlighted in the Figure to understand their definitions, ,making the definitions very hard to read. There is a lingering comment in this definition.

While we are not so sure what is meant here by “re-constructed” above, we feel that making incut and outcut explicit in def. 6 is unnecessary, because def. 6 is simply a functional composition including replace(), and incut/outcut edges are used in the definition of replace().

We have now highlighted the incut and outcut edges in fig. 6 (group by entities) and fig. 7 (group by activities), which are the two homogenous cases.

W7: Related Work:

1. It is not obvious why in Zoom the user has to have an understanding of workflow structure while this is not the case in their approach. In their approach also the user is selecting nodes and activities to eliminate (similar to selecting relevant workflow modules). Similar to Zoom, the paper considers validity of the provenance graph, except this paper is speciifc to PROV semantics.

Please see response to point W2, where we clarify that no knowledge of graph structure is necessary as we have suggested a policy-driven mechanism as a possible way of selecting nodes, which is oblivious of graph topology.

2. The distinction to compressing provenance graphs [18] is not evident. The proposed technique seems a strategic spin on the lineage by types. More clarification is needed as to who summaries are different than obfuscation or how a result produced by that technique is different than the result produced by path/extend/replace.

[18] (which, incidentally, cites our own preliminary report on this topic, ie., [20]) relies on the idea of “provenance paths”, i.e., “we construct a summary by aggregating all nodes that have the same provenance types, and likewise for edges, and by counting their occurrences”. This combines two things: (1) a mechanism to identify the provenance types, and (2) a mapping from nodes to types, which defines the aggregation. The latter consists of a set of provenance types connected by weighted edges and thus has a PROV representation, however “summaries in general are not valid [5] provenance graphs”.

Thus this approach has different intent and mechanism and its goal is not at all to preserve PROV validity, rather to capture the “essence” of a fine-grained set of provenance statements by observing regularities in the original graph.

This has been added to sec 2.4.

Other comments

Figure 7 (b) e5 should be shaded.

Corrected.