Structural summaries for visual provenance analysis

Houssem Ben Lahmar
IPVS | University of Stuttgart
Stuttgart, Germany
houssem.ben-lahmar@ipvs.uni-stuttgart.de

Melanie Herschel
IPVS | University of Stuttgart
Stuttgart, Germany
melanie.herschel@ipvs.uni-stuttgart.de

Abstract

Many systems today exist to collect provenance that describes how some data was derived. Such provenance represents useful information in many use cases, e.g., reproducibility or derivation process quality. Depending on the use case, collected provenance traces need to be explored and analyzed. Therefore, various approaches, including visual analysis approaches have been proposed. However, these typically focus on analyzing *individual* provenance traces.

We propose to create *structure-based summaries of provenance* by aggregating many provenance traces provided in W3C-PROV representation. We further describe the analysis tasks that apply on these summaries. We showcase the usefulness of structural summaries based on several use cases, when using appropriate visualization and interaction techniques.

Keywords Provenance, schema inference, visualization

1 Introduction

Various systems and approaches have been proposed to collect different types of provenance for various use cases, e.g., for the reproducibility and the debugging of complex computational processes [16]. Analyzing collected provenance allows users to get a better understanding of results and may lead to the discovery of information helpful to refine the computational processes for which provenance is tracked.

Given the wealth of available provenance, several works, including [9, 23], have focused on the visualization of provenance using a number of presentation paradigms (e.g., networks, node-link graphs). However, these visualizations typically do not scale with the increasing size of provenance, as they lead to cluttered visualizations. To not overwhelm users with visualizations of full provenance graphs, particularly in the initial phase of provenance exploration where a user just wants to get a rough understanding, abstraction of provenance traces has been studied [2, 8, 18]. The majority of these works adopt a common process where a single provenance trace is input to the proposed aggregation or summarization strategy. While these approaches succeed in reducing the complexity of a single provenance trace (generally in form of a graph) to analyze visually, they lack capabilities to abstract a *collection* of provenance traces. Note that we use *provenance trace* term to refer to a single provenance graph collected in one run of the computational processes for which provenance is tracked.

This paper proposes an end-to-end solution for structural summaries of a collection of provenance traces that serves visual provenance analysis. More precisely, we generate a structural summary graph that aggregates many provenance traces conforming the PROV-JSON[1] standard. An interactive visualization of this summary is rendered to help users perform various analysis tasks. To clarify our contribution, let us consider the following example.

Example 1.1 Assume that our goal is to define a university ranking that consolidates three well-known university rankings[2]. As the different source rankings use different criteria, the three source tables do not fully agree. Hence, consolidation requires the definition of a custom score derived from the three sources. Now, let us assume that the custom score yields some surprising results, e.g., the university "MIT" is ranked at tenth position. To better understand the working of the ranking, we compute the why-provenance of the suspicious result, as shown in Fig. 1. The provenance conforms to the PROV-JSON format. Essentially, "entity1" (the MIT result with rank 10) is derived from an "entity0" using "activity1", i.e., the custom scoring query that generated "entity1". This trace alone is not very insightful yet, in this example, it would be more interesting to find out how this trace relates to provenance traces of other (unsuspicious) results. This calls for generating traces for all results (we limit to 10 results in our example) and comparing those. However, visually comparing all provenance traces quickly becomes infeasible. ■

To simplify the comparative task motivated by the above example, we propose in this paper to infer the general structure of each trace. The individual structures are then unified in a merged structure with annotations that quantify the coverages of sub-structures in the underlying collection of traces.

Example 1.2 The structural summary of our ten sample provenance traces is shown in Fig. 2. Vertices present different structures available in the ten provenance traces. Each structure definition is depicted in a white box near each vertex.

TAPP'19, June 2019, Philadephia, PA, USA

[1] https://www.w3.org/Submission/2013/SUBM-prov-json-20130424/
[2] https://www.kaggle.com/mylesoneill/world-university-rankings

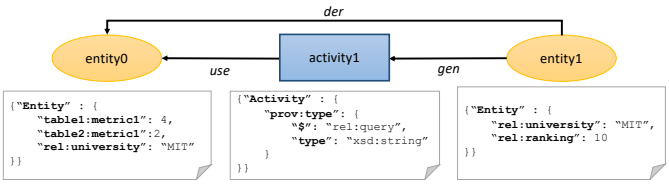

Figure 1. Provenance for MIT custom rank

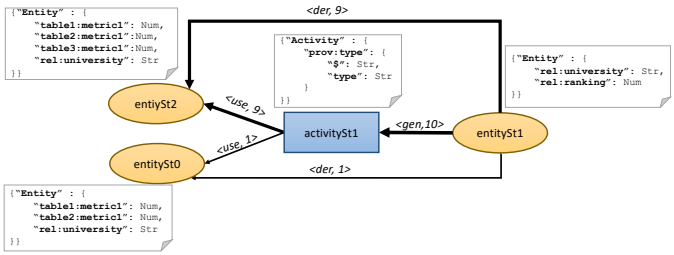

Figure 2. Structural summary of ten provenance traces

Each edge is associated with two annotations that specify the type of provenance relationship and its presence rate in the analyzed provenance traces. From this graph, we see that while most results (all conforming to the structure "entitySt1") are derived from entities conforming to the structure of "entitySt2", one entity is derived from a different structure, i.e., "entitySt0". Interaction with this summary graph allows us to identify "entitySt0" as the structure of the unexpected "MIT" result. Hence, this summary shows that "activity1" computing the consolidated ranking may be affected by an incomplete input table (i.e., no score from the third source is available). ■

The example above provides one motivating scenario that shows the usefulness of structural provenance summary graphs. Further use cases are discussed in Sec. 6. To achieve the functionality described in the example, this paper proposes a solution that infers the schema of multiple provenance traces in W3C-PROV format to then summarize those in a one graph. This approach has several benefits including (i) the summary allows to get a first rough overview about a collection of provenance traces; (ii) it is useful to highlight commonalities and differences among the traces for comparative analysis, and (iii) it allows to pinpoint corner cases and exceptions.

Overall, this paper makes the following contributions.

- **Provenance structure summarization**. We propose a two-phase algorithm to produce the structural summary of provenance traces. The first step infers the structure of each individual trace. These structures are then input to the second phase that produces the merged and weighted provenance structure summary graph, where weights translate the coverage of a relationship in the individual traces.
- **Summary visualization and analysis.** We define several visual analysis tasks over the summary graph. For selected tasks, we showcase different visualizations as part of our preliminary use-case based evaluation.
- **System implementation and evaluation.** We implement the proposed approach in a prototype, for which we provide a preliminary performance evaluation. Results show the effectiveness of our approach in terms of runtime and conciseness of inferred structures-based summaries.

The remainder of this paper is structured as follows. Sec. 2 discusses related work. Sec. 3 covers necessary definitions and formal preliminaries. We present our structure-based summary approach in Sec. 4. Visual analysis tasks and use cases are presented in Sec. 5 and Sec. 6. Finally, we discuss our implementation and evaluation in Sec. 7.

2 Related work

This section reviews research in fields related to our work, i.e., provenance summary and schema inference and integration.

2.1 Summary of provenance traces

Several strategies have been proposed to simplify a single provenance trace. One of them is the temporal-based strategy which was widely adopted by many works including [8, 23]. While this strategy reduces successfully the complexity when dealing with one provenance trace, it does not apply on a set of provenance traces collected at different periods.

Other strategies include semantic summaries [2, 12, 17]. These require expert knowledge to define semantic mappings between provenance components. The same holds for the user-defined summaries [7, 18], where sufficient user knowledge about the processed provenance trace is required to appropriately perform grouping operations.

Template-based summarization consists in merging subparts of an analyzed provenance trace when they have the same shape (template) [23]. However, the specification of these templates is left to users, i.e., they have to specify patterns they expect in their provenance traces that can be collapsed in a simplified visualization. In the same context, Moreau et al. [19] propose a parametric summary solution that compresses firstly paths of size k and then merges compressed paths having the same shapes. While this solution could be applied on a set of provenance traces, it is still unclear how to set a reasonable value of k that generates a concise summary neither too general nor too specific.

Finally, approaches such as [11, 20] declare the template subsequently collected provenance follows. This template can be seen as a static summary that is independent of the set of analyzed provenance traces. Opposed to generating a static summary, our approach considers actual provenance traces and exposes their similarities and differences. Our approach reveals also which parts of the static summary are actually covered by the set of analyzed provenance traces.

Overall, our work differs from discussed works in two main aspects: (i) most of the discussed works rely on expert knowledge to summarize provenance. This is not required using our approach; (ii) opposed to approaches generating provenance templates using a top-down approach, we follow a bottom-up strategy to infer structures from analyzed provenance traces.

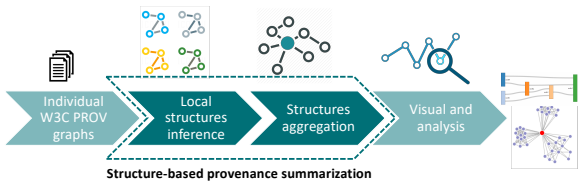

Figure 3. Overview of the proposed approach

2.2 Schema inference and schema integration

Our approach is close in spirit to schema inference, where given a set of datasets, a common, generalized schema in a predefined data model is derived. Given our focus on semi-structured W3C PROV provenance traces, our work is closely related to schema inference for XML and JSON data [4, 14].

Our work is also related to schema integration, especially for semi-structured data [10] where, given a set of schemas, a unified schema is determined. This latter covers all concepts and properties of the input schemas and correctly models the constraints defined in these input schemas.

While the above methods may complement our approach, they are left for future research. In particular, integrating such techniques requires to further investigate the information loss and associated impact on analytical applications on provenance traces. Our current focus lies on inferring a structural summary that reports primitive types of provenance components and that highlights dependencies (an important aspect for provenance analysis) between inferred structures.

3 Preliminaries and system overview

As discussed earlier, we propose a solution relying on structural aggregation for visual provenance analysis. This solution follows the pipeline shown in Fig. 3, briefly described below.

Individual W3C-PROV graphs. We assume that all input provenance traces conform to the W3C-PROV data model [6]. As defined in [18], this data model defines three types of sets: (i) Entities (*En*), i.e., data, documents; (ii) Activities (*Act*), i.e., processes, actions acting upon entities; and (iii) Agents (*Ag*), i.e., humans, software. The W3C-PROV data model further defines a set of relationships among entities, activities, and agents. Examples of relationships include:

$usage : use \subseteq Act \times En$ $\quad attribution : att \subseteq En \times Ag$

$derivation : der \subseteq En \times En$ $\quad generation : gen \subseteq En \times Act$

Note that the W3C-PROV data model has many variations. In what follows, we focus mainly on the PROV-JSON representation of the W3C-PROV given the wide adoption of JSON as an exchangeable data format. This allows us to reuse existing solutions, e.g., for JSON schema inference [4] or visualization frameworks. Furthermore, JSON is semi-structured and thus allows to easily model heterogeneous provenance traces.

A PROV-JSON provenance trace follows the tree format displayed in Fig. 4. Each information present at the top-key level maps to a W3C-PROV concept described above.

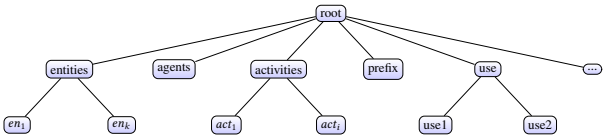

Figure 4. Top-key level of a PROV-JSON trace

The provenance traces represented in PROV-JSON format generally form a graph, that we define as follows.

Definition 3.1 (Provenance graph) *A provenance graph is a directed graph $G_P(V, R)$, where V is the set of vertices and R maps to provenance relationships. Let root $\subseteq V$ be a set of nodes with an in-degree of 0 and an out-degree larger than 0. These nodes represent entities that are derived as described by the provenance trace. Successors of root are entities $En \subseteq V$, activities $Act \subseteq V$, and agents $Ag \subseteq V$ involved in generating the entities described by root. Each edge $e = (\langle v_1, v_2\rangle, t)$, represents a provenance relation $r \subseteq R$ of type t (e.g. gen, use) between v_1 and v_2 where $\{v_1, v_2\} \subseteq V$.*

Fig. 1 depicts an example of a provenance graph obtained using Perm [13] for our university ranking example. It describes the why-provenance of the result ranking of "MIT", that is, it returns the source data relevant to produce this result. The set *root* contains a unique entity "entity1" that results from the ranking query. This latter is presented as an activity that uses the entity "entity0" to generate the result. Fig. 1 further includes white boxes associated with each node in the provenance graph. These contain the concrete provenance information recorded for each node. For instance, the provenance recorded about "entity1" includes the name of the university and the ranking value.

Structure-based provenance summarization. The provenance graphs defined above serve as input to our structure-based provenance summary approach, that consists of two steps. First, we infer individual provenance structures associated to individual provenance traces (local structures inference in Fig. 3). The second step aggregates individual structures into a single W3C-PROV compliant graph that structurally summarizes input provenance traces. More formally, we define the structure-based summary graph as follows.

Definition 3.2 (Structure-based summary graph) *A structure-based summary graph is a W3C-PROV compliant directed graph $SSG(PS_s, RS_s)$ associated to a set of provenance graphs $G=\{G_{P1}, \ldots, G_{Pn}\}$ where PS_s is the set of vertices referring to provenance structures defined later in Def. 4.1 and RS_s are edges mapping to the set of inferred provenance relationships. Each relation $r \subseteq RS_s$ is presented as $(\langle ps_1, ps_2\rangle, t, c)$ where $\langle ps_1, ps_2\rangle$ is the edge connecting two structures $ps_1, ps_2 \in PS_s$, the type t presents the provenance relationship type, and the cardinality c is the number of provenance relationships in G, that follow the structure of r.*

Fig. 2 shows an example of a structure-based summary graph. The content of this graph is discussed in Example 1.2.

$p ::= Act \mid En \mid Ag$	provenance high level types
$Act, En, Ag ::= \epsilon \mid R$	provenance types
$R ::= \{l_1 : s_1, \ldots, l_i : s_i\}$	Record type
$s ::= B \mid R \mid A$	structure types
$A ::= [s_1, \ldots, s_j]$	Array type
$B ::= null \mid Bool \mid Num \mid Str$	Basic type

Figure 5. Syntax of provenance types

Visualization and analysis. The last component of our pipeline generates visualizations suited for various visual analysis tasks that rely on the structure-based summary graph.

After this overview, we now delve into the details of individual components of our solution in Sec. 4 and Sec. 5.

4 Provenance structure inference

Fig. 4 shows the top-key level structure of a PROV-JSON provenance trace, which contains nodes referring to W3C-PROV relationships (e.g., use, gen). Those nodes have usually standard structures respecting the constraints defined in the W3C-PROV data model. In contrast, nodes mapping to entities, activities, and agents may have various structures (e.g. entities "entity0" and "entity1" in Fig. 1) depending on the underlying provenance collector. Hence, a special focus is given in what follows to the subset of top-key level's elements, called provenance components, that contains entities, activities, and agents. The provenance components follow the syntax shown in Fig. 5, extending the syntax proposed in [4] to fit the content of W3C-JSON provenance traces.

=A provenance component p has either an activity (Act), entity (En) or agent (Ag) type. These provenance high level types encompass records that are sets of pairs. Each pair has a label l associated with a structure type s. This latter could be a record, an array, or basic type. The array type is a sequence of structure types s while basic values B comprise *null* values, booleans $Bool$, numbers Num, and strings Str.

4.1 Prov-type inference

The first phase of our approach performs a type inference for each provenance component p present in the provenance trace. We adjust inference rules proposed in [4] to infer types called in what follows *prov-types* defined in Fig. 5.

Prov-type inference is done according to the inference rules in Fig. 6. We distinguish two types of inference rules: (i) those without premise: they infer the prov-type of value by simply reflecting the type of the value itself and (ii) rules with premise: they require the recursive prov-type inference of each element present in the premise to generate the global prov-type indicated in the conclusion part.

Based on inference rules presented in Fig. 6, we can define *prov-type*, the first step towards generating a structural provenance summary of a collection of provenance traces.

Definition 4.1 (Prov-type) *A prov-type for a vertex v present in $G_P(V, R)$ corresponds to the structure inferred for v.*

$$\frac{}{\vdash \{\} \rightsquigarrow \texttt{Null}}\ (\textsc{TypeEmpty}) \qquad \frac{}{\vdash true/false \rightsquigarrow \texttt{Bool}}\ (\textsc{TypeBool})$$

$$\frac{s \in Number}{\vdash s \rightsquigarrow \texttt{Num}}\ (\textsc{TypeNumber}) \qquad \frac{s \in String}{\vdash s \rightsquigarrow \texttt{Str}}\ (\textsc{TypeString})$$

$$\frac{\vdash s_1, \ldots s_n \rightsquigarrow T_1, \ldots, T_n}{\vdash [s_1, \ldots, s_n] \rightsquigarrow [T_1, \ldots, T_n]}\ (\textsc{TypeArray}) \qquad \frac{\vdash s_i \rightsquigarrow T_i \ \ \vdash s_j \rightsquigarrow T_j \ \ \forall i, j\ i \neq j \Rightarrow l_i \neq l_j}{\vdash \{l_1{:}s_i, \ldots, l_i{:}s_i\} \rightsquigarrow \{l_1{:}T_i, \ldots, l_i{:}T_i\}}\ (\textsc{TypeRec})$$

$$\frac{\vdash s \rightsquigarrow T}{\vdash Ag(s) \rightsquigarrow Agent(T)}\ (\textsc{TypeAgent}) \qquad \frac{\vdash s \rightsquigarrow T}{\vdash Act(s) \rightsquigarrow Activity(T)}\ (\textsc{TypeAct}) \qquad \frac{\vdash s \rightsquigarrow T}{\vdash En(s) \rightsquigarrow Entity(T)}\ (\textsc{TypeEntity})$$

Figure 6. Prov-type inference rules.

Following our definition, the prov-type inferred for the entity "entity1" shown in Fig. 1 is:

```
{ "Entity":
    {
        "rel:University_name": Str,
        "rel:ranking": Num
    }
}
```

4.2 Individual structural provenance graph

Once we compute the prov-type of each vertex in a provenance graph, we are able to infer the individual structural provenance graph that we define as follows.

Definition 4.2 (Individual structural provenance graph) *$GS(P_s, R_s)$ is an individual structural provenance graph associated to a provenance graph $G_P(V, R)$. It is a W3C-PROV compliant directed graph where P_s is the set of prov-types for all $v \in V$ and R_s is the set of inferred relationships structures. Each edge $e_s = (\langle ps_1, ps_2 \rangle, t, c)$, is a provenance relationship structure in R_s between two structures $\{ps_1, ps_2\} \subseteq P_s$. The edge e_s is labeled by t that refers to the type of provenance relationship and by a cardinality c computed as $c = |R_{sim}|$ such that $R_{sim} \subseteq R$ and $\forall e_r = (\langle v_1, v_2 \rangle, t') \in R_{sim}$, $e_s.t = e_r.t'$ holds and $\forall i \in \{1, 2\}, \exists j \in \{1, 2\}$ such that the prov-type of v_i is equal to ps_j.*

Note that the following properties hold for the individual structural provenance graph $GS(P_s, R_s)$:

Lemma 4.3 (Properties of structural provenance graph)

- $\forall ps, ps' \in P_s$, $ps \neq ps'$
- $\forall p_s \in P_s$, $\exists\, v \in V$ *such that its prov-type is equal to* p_s
- $\forall e_s = (\langle ps_1, ps_2 \rangle, t, c) \in R_s, \exists e = (\langle v_1, v_2 \rangle, t') \in R$ *such that* $e_s.t = e.t'$ *and* $\forall i \in \{1, 2\}, \exists j \in \{1, 2\}$ *such that the prov-type of* v_i *is equal to* ps_j.

Fig. 7 shows an example of individual structural provenance graph for the provenance graph shown in Fig. 1. It contains three nodes whose prov-types are shown in white boxes below each vertex. For instance, the node "activitySt1" has a prov-type that was inferred from the node "Activity1" present in Fig. 1. Fig. 7 depicts also cardinalities and types associated to each edge. For instance, the edge between "activitySt1" and "entitySt0" has the type "usage" and a cardinality equals to one. The cardinality is explained by the provenance

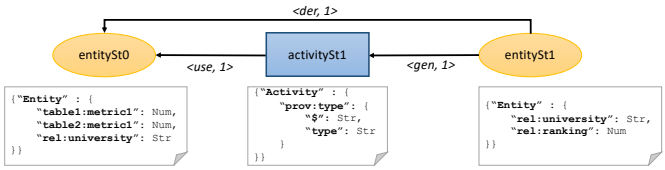

Figure 7. Structural summary of MIT's ranking provenance

Algorithm 1: Individual Prov Structure Inf (G_p)

Input: G_p : provenance graph
Output: GS: individual structural provenance graph
1 $s_{cand} \leftarrow$ an empty map of $\langle S, id \rangle$ elements, where S is the prov-type, and id is an identifier of a provenance component;
2 $ST_{inf} \leftarrow$ an empty hash-map of $\langle S, L(id) \rangle$ elements, where S is the prov-type, and $L(id)$ is the list of provenance components ids following S;
3 $REL_{Inf} \leftarrow$ an initially empty list of $\langle \{d_1..d_k\}, t, c \rangle$ elements, where t refers to the type of provenance relation, $\{d_1..d_k\}$ are related provenance components and c refers to the cardinality ;
/* Inference of provenance components structures */
4 **foreach** $e \in G_p.Entity() \cup G_p.Activity() \cup G_p.Agent()$ **do**
5 $\quad s_{cand} \leftarrow Add(\langle InferType(e), e.getId() \rangle)$;
/* Aggregation of prov-types */
6 $ST_{inf} \leftarrow AggregateTypes(S_{cand})$;
/* Inference of provenance relations structures */
7 **foreach** $r \in G_p.getRelations()$ **do**
8 $\quad r_{rel} \leftarrow r.getRelationsElements()$;
9 $\quad s_{rel} \leftarrow \emptyset$;
10 $\quad$ **foreach** $comp \in r_{rel}$ **do**
11 $\quad\quad s_{rel} \leftarrow Add(ST_{Inf}.get(comp.getID()))$;
12 $\quad REL_{Inf} \leftarrow Add(s_{rel}, r.getType, 1)$;
/* Provenance relations structures aggregation */
13 $REL_{Inf} \leftarrow AggregateRels(REL_{inf})$;
14 $GS \leftarrow$ new Graph (ST_{inf}, REL_{inf});
15 return GS;

graph shown in Fig. 1 where we have one edge of type "usage" between nodes "activity1" and "entity0".

To infer an individual structural provenance graph, we leverage the following two fundamental definitions.

Definition 4.4 (Vertex structural equality) *Given a provenance graph $G_P(V, R)$, we say that two vertices are structurally equal if they have the same prov-type.*

Definition 4.5 (Edge structural equality) *We consider that edges $(\langle el1_1, el1_2 \rangle, t_1)$ and $(\langle el2_1, el2_2 \rangle, t_2)$ are structurally equal if they have the same provenance relationship type $(t_1 = t_2)$ and $\forall i \in \{1, 2\}, \exists j \in \{1, 2\}$ such that $el1_i$ and $el2_j$ are structurally equal.*

Algorithm 1 leverages the two previous definitions to infer an individual structural provenance graph for a provenance graph G_p. Firstly, we go over the provenance components (lines 4–5) and we generate their associated prov-types using the function $InferType$ that implements rules specified in Fig. 6. Pairs of inferred structures and provenance components ids are then stored in the map s_{cand}. In line 5, we aggregate similar structures present in s_{cand} based on Def. 4.4 and we store results in ST_{inf} that contains inferred prov-type s and the full ids list of provenance components that follow s.

Afterwards, we go over provenance relationships (lines 7–12). For each relation r, we identify ids of provenance components involved in r. We resort to the ST_{inf} to replace the set of ids by their corresponding prov-types. Later, we store identified prov-types, the type of r, and a cardinality (set to one) in REL_{inf} (line 12). Finally, we aggregate the cardinality of relationships that are structurally equal using function $AggregateRels$ that implements Def. 4.5.

Once we specify the set of prov-types ST_{inf} and the set of inferred relationships REL_{inf}, we get the full image of the individual structural provenance graph of G_p.

4.3 Structure-based summary graph generation

At this stage, our goal is to further aggregate the set of individual structural provenance graphs returned by the previous step in order to generate a unique structure-based summary.

We choose to apply an exact merge to fuse input structural summary graphs. For that, we leverage again Def. 4.4 and Def. 4.5 to merge vertices and edges of individual structural graphs that are structurally equal.

Our choice of exact merge is justified by its capability to mitigate the loss of information by providing exact structures available in a set of provenance traces. This is not the case for the inexact merge that fuses nodes having different yet compatible structures. While our approach could be extended to support the inexact merge, we prefer to leave it for future research to assess or convey the impact of inexact merge techniques on visual analytics tasks.

Based on the current implementation of merge, we can redefine the structure-based summary graph as follows.

Lemma 4.6 (Structure-based summary graph properties) *The structure-based summary graph $SSG(PS_s, RS_s) = \bigcup_{i=1}^{n} GS_i(P_{si}, R_{si})$ where GS_i are individual structural provenance graphs. It has a surjective map M: $\bigcup_{i=1}^{n} P_{si} \longrightarrow PS_s$ such that:*

- $\forall p \in P_{si}, \exists p^* \in PS_s$ *with p and p^* are structurally equal*
- $\forall r \in R_{si}, \exists\, r^* \in RS_s$ *with r and r^* are structurally equal*

While it is possible to incrementally perform a set of pairwise merges of individual structural provenance graphs to construct the structure-based summary graph, we choose to design the structure-based summary inference using a map-reduce model to ensure the capability of processing large collections of provenance traces. We justify our choice of this model by the soundness property already proved for the structure inference approach [4]. Our approach enjoys also commutativity and associativity properties as we use an exact matching strategy to aggregate structures. These are important properties since the map-reduce model may arbitrarily split the input set of provenance traces. We postpone the discussion of the performance study of our approach to the Sec. 6.

5 Visual analysis of summary graphs

Based on the structure-based summary graph for a collection of provenance traces obtained as described in the previous section, we define several visual analysis tasks that apply

on this kind of summary. For each task, we also evoke visualization techniques that potentially fit the task.

Ⓐ *High level overview* The primary goal of our approach is to generate a structural summary that is easy to read and that allows analysts to easily grasp which different structures are present in the provenance traces. To avoid overwhelming analysts with too many details at an early stage of their analysis, the visualization should be limited to the rendering of basic information such as vertices and edges representative of structures and provenance relationships.

Ⓑ *Interactive visual analysis* Rendering a simple visualization of structure-based summary facilitates the analysis task. Further information can be offered on demand using interaction e.g., hovering over nodes or edges.

Ⓒ *Visual comparison* Possible visual analysis tasks include the comparison between the summary and provenance instances. Here, an analyst may compare a particular provenance trace to the inferred structural summary graph. Brushing and linking interaction techniques could be employed in this task to render/highlight structures and their corresponding provenance traces.

Ⓓ *Homogeneity overview* Our approach assigns cardinalities to edges present in the provenance structural summary graph. This information should be communicated clearly as it serves to investigate the homogeneity/heterogeneity of analyzed provenance traces. Sankey diagrams [22] are a candidate visualization for this task, given their ability to render edges with various width expressing the importance of cardinalities. We can also resort to the change of color encodings to highlight "outliers", e..g, in Example 1.1 where some edges have low cardinalities in comparison to the remaining edges in the summary graph.

Ⓔ *Visual identification of patterns* Using cardinality information, we can reveal recurrent patterns among the set of analyzed provenance traces. The visualization needs to highlight sub-graphs having high cardinalities compared to other information present in the summary graph. Sankey diagram [22] could be used also for this analysis task.

Ⓕ *Visualization of dense regions of the summary* Structure-based summary graphs may contain dense regions where vertices are highly connected. This specific range of nodes may be subject of bottleneck or may present a heavily shared component. Note that the presence of a high connectivity can easily lead to a cluttered visualization. To avoid that, we can use force-directed graphs that reduce edge crossings by making edges repel each other.

The list of proposed visual analysis tasks and their visualizations is not exhaustive. Indeed, a thorough investigation of other possible visual analysis tasks is left for future research.

6 Case study

In this section, we illustrate the feasibility of some visual analysis tasks discussed in Sec. 5 over various scenarios.

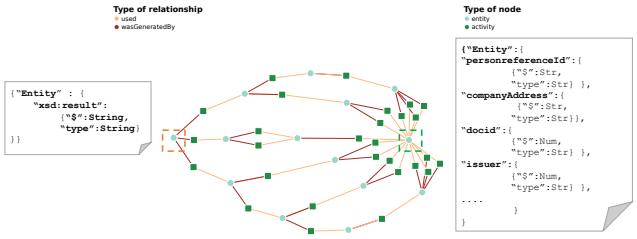

Figure 8. Structural summary for HIL provenance graphs

Alongside these scenarios, we provide a set of structure-based summary graph visualizations that are available online [1].

6.1 Visual debugging of a tracked process

Referring back to Example 1.1, our ranking is the result of an SQL query that combines rankings from three tables using a full outer join operation (on the name of the university) to generate the final scores of universities.

Our structure-based summary graph, shown in Fig. 2 (and available online [1]) summarizes the ten why-provenance traces associated with the top-ten query results. As explained in Example 1.2, the inspection of this graph (Task Ⓐ) reveals that "activitySt1" has a doubtful relationship of type "usage" with the structure "entitySt0" (Task Ⓓ).

Indeed, it turns out that the use of a full outer join in our ranking query is unreliable as it tolerates the integration of incomplete information. Specifically, MIT university has a different name in one of the three ranking tables. This entails the presence of a tuple resulting from joining two ranking tables, used later to compute the final score of MIT university.

6.2 Data integration flow transparency

Our second use case considers a data integration process specified using the high-level integration language (HIL) [15]. The sample integration flow extracts information about key people of the US financial sector. It takes a set of reports generated by companies to construct a set of individual reports about persons' careers. HIL was recently instrumented to capture provenance [21], allowing us to generate five provenance traces tracking the discussed flow when integrating information about five persons. These traces, converted to PROV-JSON format, are summarized using our approach.

As shown in Fig. 8, we render this summary using a force-directed graph (Task Ⓐ) given its capacity to place in convenient way vertices and edges by assigning forces to them.

The inspection of Fig. 8 reveals two important information. Firstly, we can figure out that the structure highlighted using an orange dashed box corresponds to the outcome of the implemented data integration flow as it is the only structure having only relationships of type "generation" with three activities structures. This reveals also that this particular structure was populated by three integration sub-flows (Task Ⓔ). By tracing back interactively these sub-flows, we can learn more about the implemented data integration flow.

pattern	clause
claim	wasGeneratedBy(e_1, a_1) with e_1 ={"foaf:name":{"$":"file.txt", "type":"string"},"prov:type":{"$": "kimlab","type":"qualified_name" }}, a_1 : {"kimlab:htseq-stranded": {"$": "gencode2","type":"string"}}
confirmation pattern	wasGeneratedBy(e_1, a_1) with e_1 = {"foaf:name": {"$": *,"type":*}, "prov:type":{"$": *,"type":* }}, a_1 = {"kimlab:htseq-stranded":{"$": *,"type":*}}
witness pattern	wasGeneratedBy(e_1, a_1) with e_1 = {*}, a_1 = {*}

Table 1. Clauses used in corroboration process

Furthermore, Fig. 8 shows a second important finding. Indeed, all data integration sub-processes stem from the single structure highlighted by a green dashed box (Task Ⓕ). This later presents the structure of input reports used by the tracked data integration process. The input reports contain personal information about the key people including their address, their positions, as well as information related to the reports such as the issuer (the editor) of a report and the ID of the issuer.

6.3 Corroboration

Our final use case comes from the life science domain relating to Next Generation Sequencing (NGS) and is inspired by [3]. This project presents a workflow including six possible analysis stages that are invoked differently depending on the version of the workflow. One major problem already mentioned in [3] is the lack of NGS workflow transparency. Tackling this problem resulted in the collection of more than 800 provenance traces, which are publicly available[3].

We assume that an analyst is working on this collection to study impacts of analysis stages on the generated results. Given the large size of this collection, the analyst randomly picks some provenance traces whose analysis reveals the presence of common information presented in the first row of Tab. 1. This clause states that an analysis stage called "HTSeq" presented as an activity a_1 is always involved in the generation of some intermediate results. As it is tedious to check all provenance traces, *the analyst claims that data input to the tracked workflow is necessarily processed by the analysis stage "HTSeq"*.

To assess the truthfulness of the analyst's claim, we resort to the approach proposed in [5]. It extracts the set of confirmation patterns (witnesses confirming to the structure of the claim) and the set of witness patterns (generic version of the claim) from the existing provenance traces. The second and third rows of Tab. 1 describe these two sets, which are finally compared to compute the reliability of the claim.

Note that clauses of confirmation and witness patterns could be easily inferred using our approach. To this end, we generate the structure-based summary graph representative of provenance traces available in the analyzed repository.

Fig. 9 depicts an excerpt of the structure-based summary graph. We choose to only render the set of structural relationships of type "generation" since they map to the witnesses set

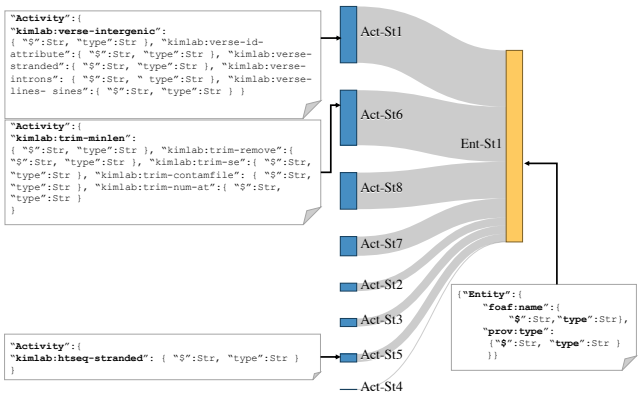

Figure 9. Excerpt of structural provenance summary graph

specified in the third row of Tab. 1 (Task Ⓔ). For that, we use a Sankey diagram as we need to highlight the cardinality of inferred relationships' structures that will be used to estimate the reliability of the claim. Indeed, the width of edges in the Sankey diagram corresponds to the cardinality value of inferred relationships. For instance, we can easily find the set of confirmation patterns (confirming to the second row of Tab. 1) which corresponds to the edge between nodes "ActSt5" and "EntSt1". The cardinality of this relationship is not high given the mediocre width of this particular edge. Hence, we can get a rough idea about the reliability of the claim (Task Ⓓ) which seems not high as the number of confirming patterns is significantly less than the number of witnesses (sum of all edges). We can also learn interactively structures and exact cardinalities, that are used to compute the exact value of the reliability of the claim following formulas proposed in [5].

7 Performance evaluation

We have studied firstly the conciseness of summary graphs. To this end, we considered provenance traces of use cases $UC1$, $UC2$, $UC3$, referring to the three case studies discussed in Sec. 6. For each use case, we collected ten provenance traces whose graphs information are summarized in Tab. 2.

The results of these experiments are shown in Fig. 10. First, we have studied the conciseness of structure-based summary graphs generated using our approach from three provenance collections of $UC1$, $UC2$ and $UC3$. For each provenance set, we generate its structure-based summary. Later, we compare the number of inferred structures (of type entity, activity and relations) with the number of entities, activities, and relations in the input provenance traces set. As shown in the left part of Fig. 10, our approach performs a high simplification rate above 80%) for the three studied use cases. This indicates

Use case	avg(#activities)	avg(#entities)	avg(#relations)
$UC1$	1	11.4	21.8
$UC2$	114.2	115.3	228.6
$UC3$	5	6	23

Table 2. Overview of processed provenance traces

[3]https://github.com/alawinia/provClustering/

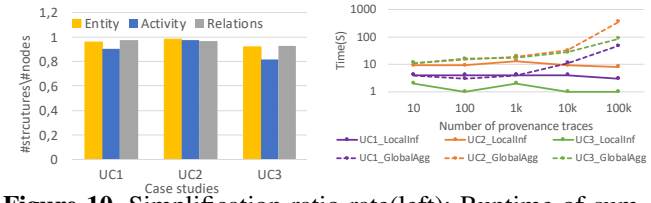

Figure 10. Simplification ratio rate(left); Runtime of summary computation steps(right)

that structure-based summary graphs allow producing concise summaries of collections of provenance traces.

We have also studied the runtime of our approach implemented using a map-reduce model. Thus, we implemented a provenance generator that takes the provenance traces available in $UC1$, $UC2$, and $UC3$ and generates variants according to the structure of each seed. This results in several synthetic provenance trace sets of varying sizes, that are summarized later using our approach. We performed our experiments on a cluster containing 3 nodes, each containing 6 cores and 256GB of RAM. During experiments, we measured runtimes of two main steps of our approach (shown in Fig. 3): (i) $LocalInf$ maps to the local inference of structures and (ii) $GlobalAgg$ concerns the aggregation of structures. The right side of Fig. 10 reports the runtimes of the two steps on a logarithmic scale over various sizes of provenance trace collections. The two steps have initially similar runtimes when processing small provenance traces sets. Yet, the gap between the two steps increases clearly with the number of processed provenance traces. Indeed, as the number of provenance traces increases, the size of the hash map storing inferred structures and their associated objects increases. Later, hash map values storing large lists, are accessed during the global aggregation to group edges that are structurally equal. This is costly and leads to an increase of global aggregation runtime.

8 Summary and future work

This paper presents a solution that infers a structure-based summary from a set of provenance traces available in PROV-JSON format. We discuss our proposed solution as well as possible visual analytics tasks that may be applied on this type of summary. Furthermore, we provide some illustrative use cases that showcase the utility of this kind of summaries. We implement also our approach using a map-reduce fashion. Our preliminary experimental evaluation studies both the runtime and conciseness of summarization. Several points for future research have been mentioned throughout the paper, including the integration of schema inference and schema integration techniques to our approach as well as the study of different visualizations for different analytical tasks.

Acknowledgments

This research is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Projektnummer 251654672 – TRR 161

References

[1] 2019. Case study. (2019). https://www.ipvs.uni-stuttgart.de/abteilungen/de/forschung/projekte/StructSum/examples.html
[2] E. Ainy, P. Bourhis, S. Davidson, D. Deutch, and T. Milo. 2015. Approximated Summarization of Data Provenance. In *CIKM*.
[3] A. Alawini, L. Chen, S. Davidson, S. Fisher, and J. Kim. 2018. Discovering Similar Workflows via Provenance Clustering: A Case Study. In *IPAW*.
[4] M. A. Baazizi, H. Ben Lahmar, D. Colazzo, G. Ghelli, and C. Sartiani. 2017. Schema Inference for Massive JSON Datasets. In *EDBT*.
[5] L. Barakat, P. Taylor, N. Griffiths, and S. Miles. 2017. Corroboration via Provenance Patterns. In *TaPP*.
[6] K. Belhajjame, R. B'Far, J. Cheney, S. Coppens, S. Cresswell, Y. Gil, P. Groth, G. Klyne, T. Lebo, J. McCusker, S. Miles, J. Myers, S. Sahoo, and C. Tilmes. 2012. *PROV-DM: The PROV Data Model*. Technical Report. http://www.w3.org/TR/prov-dm/
[7] O. Biton, S. C. Boulakia, and S. B. Davidson. 2007. Zoom*UserViews: Querying Relevant Provenance in Workflow Systems. In *VLDB*.
[8] M. A. Borkin, C. S. Yeh, M. Boyd, P. Macko, K. Z. Gajos, M. Seltzer, and H. Pfister. 2013. Evaluation of filesystem provenance visualization tools. *TVCG* 19 (2013), 129–144.
[9] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, T. Vo, and H. T. Silva. 2006. VisTrails : Visualization meets Data Management. In *SIGMOD*.
[10] L. Chiticariu, P. G. Kolaitis, and L. Popa. 2008. Interactive generation of integrated schemas. In *SIGMOD*.
[11] V. Curcin, E. Fairweather, R. Danger, and D. Corrigan. 2017. Templates As a Method for Implementing Data Provenance in Decision Support Systems. *J. of Biomedical Informatics* 65 (2017), 1–21.
[12] W. M. de Oliveira, P. Missier, K. Ocaña, D. de Oliveira, and V. Braganholo. 2016. Analyzing Provenance Across Heterogeneous Provenance Graphs. In *IPAW*.
[13] B. Glavic and G. Alonso. 2009. The Perm Provenance Management System in Action. In *SIGMOD*.
[14] J. Hegewald, F. Naumann, and M. Weis. 2006. XStruct: Efficient Schema Extraction from Multiple and Large XML Documents. In *ICDE*.
[15] M. A. Hernández, G. Koutrika, R. Krishnamurthy, L. Popa, and R. Wisnesky. 2013. HIL: a high-level scripting language for entity integration. In *EDBT*.
[16] M. Herschel, R. Diestelkaemper, and H. Ben Lahmar. 2017. A survey on provenance - What for? What form? What from? *International Journal on Very Large Data Bases* 26 (2017), 881–906.
[17] D. Koop, J. Freire, and C. T. Silva. 2013. Visual summaries for graph collections. In *PacificVis*.
[18] P. Missier, J. Bryans, C. Gamble, V. Curcin, and R. Dánger. 2014. ProvAbs: Model, Policy, and Tooling for Abstracting PROV Graphs. In *IPAW*.
[19] L. Moreau. 2015. Aggregation by Provenance Types: A Technique for Summarising Provenance Graphs. In *ETAPS*.
[20] L. Moreau, B. V. Batlajery, T. D. Huynh, D. Michaelides, and H. Packer. 2018. A Templating System to Generate Provenance. *TSE* 44 (2018), 103–121.
[21] S. Oppold and M. Herschel. 2018. Provenance for Entity Resolution. In *IPAW*.
[22] P. Riehmann, M. Hanfler, and B. Froehlich. 2005. Interactive Sankey diagrams. In *IEEE INFOVIS*.
[23] H. Stitz, S. Luger, M. Streit, and N. Gehlenborg. 2016. AVOCADO: Visualization of Workflow-Derived Data Provenance for Reproducible Biomedical Research. *Comput. Graph. Forum* 35 (2016), 481–490.