Approximated Summarization of Data Provenance

Eleanor Ainy
Tel Aviv University
eleanora@mail.tau.ac.il

Pierre Bourhis
CNRS CRIsTAL UMR 9189
pierre.bourhis@univ-lille1.fr

Susan B. Davidson
University of Pennsylvania
susan@cis.upenn.edu

Daniel Deutch
Tel Aviv University
danielde@post.tau.ac.il

Tova Milo
Tel Aviv University
milo@post.tau.ac.il

ABSTRACT

Many modern applications involve collecting large amounts of data from multiple sources, and then aggregating and manipulating it in intricate ways. The complexity of such applications, combined with the size of the collected data, makes it difficult to understand how the resulting information was derived. *Data provenance* has proven helpful in this respect, however, maintaining and presenting the full and exact provenance information may be infeasible due to its size and complexity. We therefore introduce the notion of *approximated summarized provenance*, which provides a compact representation of the provenance at the possible cost of information loss. Based on this notion, we present a novel provenance summarization algorithm which, based on the semantics of the underlying data and the intended use of provenance, outputs a summary of the input provenance. Experiments measure the conciseness and accuracy of the resulting provenance summaries, and improvement in provenance usage time.

Categories and Subject Descriptors

H.2 [**Database Management**]: Miscellaneous; I.1.1 [**Symbolic and Algebraic Manipulation**]: Expressions and Their Representation—*representations (general and polynomial), simplification of expressions*

Keywords

Provenance;Provisioning;Crowd-sourcing applications

1. INTRODUCTION

Complex applications that collect, store and aggregate large-scale data, and interact with a large number of users, are found in a wide variety of domains. Notable examples are *crowd-sourcing applications* such as Wikipedia, social tagging systems for images, traffic information aggregators such as Waze, or hotel and movie ratings such as TripAdvisor and IMDb.

In the context of such applications, several questions arise related to *how data was derived*. As a user of the information, what is the basis for trusting it? How do contributions vary among crowd members based on characteristics such as age or gender? If some contribution seems wrong, how does the information change if we discard it? These questions are fundamentally important to better understand the application and its results.

At its core, the answer to these questions is based on the *provenance* of the collected data and resulting information, that is, *who* provided the data in *what* context and *how* the information was derived. However, provenance goes well beyond simply providing a log of the application execution. In particular, the algebraic model of provenance based on semirings of [21, 7] can be used to *explain* results by correlating input with output data, and tracking important details of the computational process that took place. As shown in [17], it can also be used to *provision* the result with respect to hypothetical scenarios, i.e. to observe changes to the result based on changes to the input without re-running the process. Detailed tracking of provenance is therefore an essential vehicle for the applications mentioned above.

As an example, consider a crowd-sourced application for movie reviews, where the number of movies, and number of reviews for each movie, may be very large. An *aggregated score* for each movie is computed by combining the scores of multiple different users, possibly accounting for their previous reviews and for their preferences. These features, and the way in which they are used in the computation, should all be reflected in the provenance. In turn, this provenance may be presented to explain results such as the computed recommendations of movies, or to provision them, e.g. to determine how the average movie rating would change if we ignore ratings by some group of users.

Unfortunately, the large amount of data and complexity of processing the data means that the resulting detailed provenance information can be overwhelming. Presenting it in full, as an explanation for a computation, may make it extremely hard for users to understand. In this paper we therefore introduce the notion of *approximated summarized provenance*, which provides a compact representation of provenance at the possible cost of information loss. This compact representation will enable the user to see trends, for example that women aged 20-25 tended to rate a particular movie more highly than men aged 20-25.

CIKM'15, October 19–23, 2015, Melbourne, VIC, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806429.

Contributions. We present a novel algorithm that provides approximated summarization of provenance information for complex applications. The summarization is based in part on the semantics of the underlying data (such as gender, age or occupation of users), where annotations of "similar" data items are intuitively more amenable to be grouped together. More importantly, it is also geared towards the intended use of provenance (namely explanation and/or provisioning): we define a distance function between provenance expressions that is based on the intended use, and optimizing this distance while still obtaining small expressions guides the summarization. We have conducted experiments with three datasets - MovieLens, Wikipedia and DDP (Data Dependent Process), in which we compared our algorithm to other approaches and showed that our approach gives better summarizations in terms of distance and size.

Paper Organization. The rest of the paper is organized as follows. Section 2 describes workflow provenance and the provenance model. Section 3 describes the notion of provenance summarization through mappings and the quality measurements for such summarization. In section 4 we present a few propositions that, combined together, lead to our summarization algorithm. We end this section with an example of the full algorithm flow. Section 5 describes the datasets we used and also includes interesting use cases. We later describe our experimental results in section 6. Finally, related work and conclusions are discussed in Section 7.

2. MODEL DESCRIPTION

We now give an overview of the semiring provenance model of [21], and its extension to queries with aggregates in [8, 7]. This will serve as the basis for our work.

2.1 Workflows

We capture applications logic by a standard notion of workflows. One possible model for workflow [15, 7] consists of a specification and an associated set of executions. The specification can be thought of as an FSM (Finite State Machine), in which modules represent processing steps and edges indicate potential dataflow between the output port of one module to the input port of another module. In the model of [7], the workflow operates in the context of some global persistent state, i.e. some underlying database. Modules may be atomic, meaning that they are a query on the inputs to the module as well as the underlying database. Modules can also update the underlying database. A workflow *execution* (or "run") is a repeated application of modules, which are ordered according to the workflow specification. [1]

EXAMPLE 2.1. *Consider a movie rating application, in which users rate movies and the ratings are aggregated using the application logic described by the workflow in Figure 1.*

Certain information about users is known, such as gender and type (movie critic, director, audience, etc.), and stored in the Users *table in the underlying database.*

Reviews are collected by different reviewing modules, which crawl different reviewing platforms such as IMDb and newspaper web-sites. Each such module updates statistics in the Stats *table in the underlying database, e.g. how many reviews the user has submitted (*NumRate*), what their max score is (*MaxRate*), etc. (alternatively, we could use sum or any other aggregation function). A reviewing module also consults* Stats *to output a* sanitized review *by implementing some logic. The sanitized reviews are then fed to an aggregator, which computes an aggregate movies scores.*

There are many plausible logics for the reviewing modules; we exemplify one in which each module sanitizes the reviews by joining the users and statistics relations (depending on the module), keeping only reviews of users listed under the corresponding role (audience/critic) and who are "active", i.e. who have submitted more than 2 reviews. The aggregator combines the reviews obtained from all modules to compute overall movie ratings (num, max).

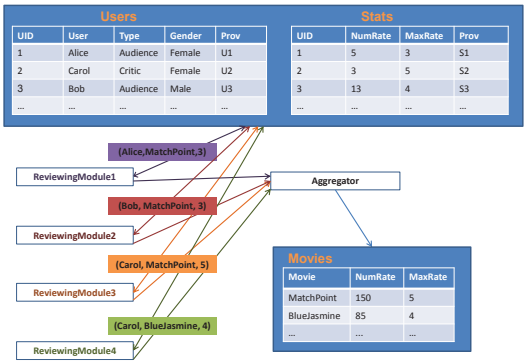


Figure 1: Example Workflow

2.2 Provenance Model

We next explain in general what a provenance model is and then use examples to illustrate the concepts described. We start by fixing a finite set *Ann* of *provenance annotations*, corresponding to the basic units of data manipulated by the application, and which can be thought of as abstract variables identifying the data. Depending on the application, these annotations may correspond to different tuples in a database, to different users, to different questions, etc.

A correspondence between data manipulation and algebraic operations in the structure of a *commutative semiring* can then be defined. A *commutative semiring* is a structure $(K, +_K, \cdot_K, 0_K, 1_K)$ where $(K, +_K, 0_K)$ and $(K, \cdot_K, 1_K)$ are commutative monoids. This means that the operations are associative and commutative with 0 and 1 standing for the neutral elements for addition and multiplication, respectively. In addition, $\cdot_K$ is distributive over $+_K$, and $a \cdot_K 0_K = 0 \cdot_K a = 0_K$.

Given our set *Ann* of basic provenance annotations, the *provenance semiring* is the semiring of polynomials with natural coefficients, with indeterminates from the set *Ann*. It is denoted $(N[Ann], +, \cdot, 0, 1)$, and was shown in [21] to capture provenance for positive relational queries. Intuitively, the $+$ operation corresponds to the *alternative use* of data (as in union and projection) and $\cdot$ to the *joint use* of data (as in join); 1 annotates data that is present, and 0 annotates data that is absent.

To capture aggregate queries, K-relations were further generalized by extending their data domain with *aggregated values* [8]. In this extended framework, relations have provenance also as part of their values, rather than just in the tuple annotations. Such a value is a *formal sum* $\sum_i t_i \otimes v_i$,

[1]This departs from the representation of executions and their provenance as multigraphs in [15]

where v_i is the value of the aggregated attribute in the i^{th} tuple, while t_i is the provenance of that tuple. We can think of $\otimes$ as an operation that pairs values (from a monoid M) with provenance annotations. Each such pair is called a *tensor*. The formal sum, presented by the $\oplus$ operation is used to capture the aggregation function.

In [8, 17] the framework was also used to define provenance for *nested aggregates* and *negation* by introducing equation and inequality elements. Intuitively an equation such as $[(d_1 \cdot d_2) \otimes m > 3]$ is kept as an abstract token and can be used in conjunction with other semiring elements. Given concrete values for d_1, d_2 and m one may test the truth value of the equality and replace the equation by the truth value [2]. A precise algebraic treatment of aggregated values and the equivalence laws that govern them is based on semimodules and is described in [8]. We will focus, for simplicity, on the case where the values monoid M is that of real numbers with numbers addition and 0.

EXAMPLE 2.2. *The basic provenance annotation set Ann consists here of* $U_1, ..., S_1, ...$
The provenance-aware value stored as MaxRate *in the aggregator's output table, the Movies table, for the "MatchPoint" tuple would be:*

$$\boldsymbol{P} = U_1 \cdot [S_1 \cdot U_1 \otimes 5 > 2] \otimes (3,1) \oplus \\ U_2 \cdot [S_2 \cdot U_2 \otimes 3 > 2] \otimes (5,1) \oplus \\ U_3 \cdot [S_3 \cdot U_3 \otimes 13 > 2] \otimes (3,1) \oplus ...$$

where U_i is a user identifier, S_i is the provenance of the user's Stats tuple, and as aggregation we use a monoid of pairs to capture the aggregated rating (MAX with value 3 in the first tensor) and how many users contributed to this value (1 per tensor here but we will next show examples with other values). Intuitively, each rating is associated with the provenance of the tuple obtained as the output of the reviewing module, namely the U_i annotation identifying the user. Each such sub-expression is multiplied by an inequality term serving as a conditional guard, indicating that the number of reviews recorded for the user is above the threshold of 2. Applying aggregation then results in coupling values (numeric reviews) with annotations to form the expression above.

2.3 Valuations and Provisioning

An important use of semiring provenance is for *provisioning*, i.e. examining changes to the application's execution that are the result of some hypothetical modifications to the data (e.g. "How would the movie ratings change if we ignore some reviews suspected as spam?"). This is formalized in [21] through the notion of *truth valuations* applied to annotations. Intuitively, specifying that U_1 is a spammer corresponds to mapping it to $false$ (and that U_1 is reliable to mapping it to $true$), and recomputing the derived value w.r.t this valuation. Such valuation can again be extended in the standard way to a valuation $V : N[Ann] \mapsto \{true, false\}$ using the following intuitive rules: (1) $0 \otimes m$ is interpreted as 0; (2) $1 \otimes m$ is interpreted as m; and (3) A comparison expression is interpreted as 1 if satisfied and as 0 otherwise.

Note that given a truth valuation for annotations, we obtain a real number for the expression by simply performing the substitution as defined above, and applying the basic semiring axioms [3].

[2]The obtained semiring is denoted by K^M in [8]. For simplicity we will abuse notation here and just use K

[3]Similarly, the provenance can capture scores of multiple movies and valuation then leads to a vector of values.

EXAMPLE 2.3. *Consider the provenance expression P of Example 2.2 and partial truth valuation that maps S_1 to 0 and U_1 to 1. Then $U_1 \cdot [S_1 \cdot U_1 \otimes 5 > 2] \otimes (3,1)$ maps to $0 \otimes (3,1) \equiv 0$: Although U_1 is mapped to 1, $S_1 \cdot U_1 \otimes 5$ is mapped to 0 and so the inequality does not hold, and the inequality expression is mapped to 0. In contrast, if S_1 is mapped to 1 then the condition would hold and we would have $(1 \cdot 1) \otimes (3,1) \equiv 3$ (notice that $(3,1) \equiv 3$ since we apply aggregation on a single user with score 3). Intuitively the second case corresponds to keeping the review, while the first one corresponds to discarding it.*

3. PROVENANCE SUMMARIZATION

The provenance model described in the previous section provides full documentation of the transformations that took place. Since the resulting expression may be extremely long and complex, we would like to summarize the provenance expression, at the possible cost of information loss. We start by formalizing summarization through a notion of *mappings*, and then discuss how to quantify the quality of a summary.

3.1 Summarization Through Mappings

Let Ann be a domain of annotations (for the $N[Ann]$ semiring) and Ann' be a domain of annotation summaries. Typically, we expect that $| Ann' | << | Ann |$. We then define a mapping $h : Ann \mapsto Ann'$ which maps each annotation to a corresponding summary. Abusing notation, this extends naturally to a homomorphism $h : N[Ann] \mapsto N[Ann]'$, i.e. define $h(a+b) = h(a) + h(b)$ and $h(a \cdot b) = h(a) \cdot h(b)$. This further extends to $N[Ann]' \otimes M$ by the standard construction $h(k \otimes m) = h(k) \otimes m$. Essentially, to apply h to a provenance expression p, each occurrence of $a \in Ann$ in p is replaced by $h(a)$. The mapped expression, $h(p)$, is a summary of the real provenance, in the sense that we lose track of some exact annotations and summarize the provenance using the abstract annotations in Ann'.

EXAMPLE 3.1. *Consider the provenance-aware expression P obtained in Example 2.2. To simplify the example we focus on the reviews of users U_1, U_2, U_3 for the movie "Match Point", and map all S_i annotations to 1 so we can discard the inequality terms. We thus obtain a simplified version of the provenance expression P:*

$$\boldsymbol{P_s} = U_1 \otimes (3,1) \oplus U_2 \otimes (5,1) \oplus U_3 \otimes (3,1)$$

Next, we map user annotations to annotation summaries that intuitively reflect values of attributes of the corresponding users. Mapping U_1 and U_2 to an annotation summary called "Female" [4], and applying congruences in the tensor structure, we obtain an expression that includes a maximum score of 5 collected from two female users:

$$\boldsymbol{P'_s} = Female \otimes (5,2) \oplus U_3 \otimes (3,1)$$

Another summary results from mapping annotations U_1 and U_3 to the annotation "Audience":

$$\boldsymbol{P''_s} = Audience \otimes (3,2) \oplus U_2 \otimes (5,1)$$

In the example, we used two possible mappings h that combine reviews based on gender or role. In general there may be many possible mappings and the challenge is, given a provenance expression p, to (a) define what a good mapping h is (correspondingly, what is a good summary $h(p)$), and (b) find such good h.

[4]We later describe which mappings are possible and which are preferable to ours.

3.2 Quantifying Summary Quality

Several, possibly competing, considerations need to be combined in quantifying the quality of a summary.

Provenance size. Since the goal of summarization is to reduce the provenance size (measured as the number of annotations), it is natural to use the size of the obtained expression (after simplifications) as a measure of its quality.

Semantic Constraints. The provenance expression obtained may be of little use if it is constructed by identifying multiple unrelated annotations. It is thus natural to impose constraints on which annotations may be grouped together. One simple example of such a constraint is to allow two annotations $a, b \in Ann$ to be mapped to the same annotation in Ann' (or to 0 or 1) only if they annotate tuples in the *same input table*, intuitively meaning that they belong to the same domain. Other constraints may be specified in the form of *taxonomies*, where available. Taxonomies give semantic relations between the underlying objects (users, movie, etc.), and are used to constrain homomorphisms by requiring that all annotations that are grouped together by mapping to the same annotation share a common ancestor.

Additional constraints involve restricting mappings based on the original input data, by requiring that annotations that are mapped together reference tuples that share values in some (or one of some) specified attributes. For example, we may specify that users that are grouped together must share a common attribute out of gender, age group, etc. This allows us to give a meaningful name to the new annotation for presentation purposes, based on the joint attribute.

Taxonomic information may also be useful for deciding between choices of mappings, and may be incorporated as part of the computation. For example, we may take into account the taxonomic distance between annotations and the annotation they are mapped to by using the MAX or SUM of these distances, and prefer mappings of annotations to a new annotation that is relatively close to them (e.g. mapping user annotations to annotation 'Guitarist' is preferable to mapping them to annotation 'Person').

Distance. Depending on the *intended use* of provenance, we may *quantify the distance* between the original and summary expressions. For that we again use the notion of valuations, and define distances with respect to a set $\mathcal{V}_{Ann}$ of valuations to the original annotations Ann.

EXAMPLE 3.2. *Consider a distance function designed to use provenance for provisioning in the presence of spammers. To simplify the example, we assume that there is a single spammer. In this case, the class of valuations considered consists of those assigning* 0 *to a single user annotation, and* 1 *to all others. A concrete aggregated value for each movie may then be computed by simply canceling every summand in which the mapped annotation is* 0*, and taking the aggregate values for the rest. (We use here the congruences* $0 \otimes m \equiv 0$*,* $1 \otimes m \equiv m$*, and the ability to embed the result in* M*, see [8].)*

A central issue is how we transform a valuation in $\mathcal{V}_{Ann}$, on the original annotations to one in $\mathcal{V}_{Ann'}$, on the new annotation summaries. We propose that this will be given by a combiner function ϕ that sets a boolean value to $a' \in Ann'$ based on the truth values assigned to a annotations that were mapped to it. For example, if ϕ is a disjunction of the truth values, then intuitively an annotation summary is cancelled only if all of the annotations it summarizes are cancelled. More formally, let Ann, Ann' be two domains of annotations and let $h : Ann \mapsto Ann'$. Further let $\phi : Ann' \mapsto N[Ann]$ be a function such that for every $a' \in Ann'$, $\phi(a')$ is a polynomial *only* in elements of $\{a \in Ann \mid h(a) = a'\}$. In a sense, ϕ complements h, by specifying how the elements that are mapped to an annotation a' should be combined. Now, any valuation $v_{Ann} \in \mathcal{V}_{Ann}$ uniquely extends to a valuation $v_{Ann'} \in \mathcal{V}_{Ann'}$ by defining $v_{Ann'}(a') = v_{Ann}(\phi(a'))$; note that the use of v_{Ann} refers to its extension to the domain of $N[Ann]$. We use $v^{h,\phi}$ to denote the valuation obtained in such a way from a valuation v and mappings h, ϕ.

We next define the distance between a provenance expression p and its summary $h(p)$ as an average over all truth valuations, of some property of p, $h(p)$, and the valuation. This property is based on yet another function we call `VAL-FUNC`, whose choice depends on the intended provenance use. We only require that it is a computable function fed as an additional input to the algorithm.

DEFINITION 3.3. *Let p be a provenance expression over a set of annotations Ann, and let $p' = h(p)$, we define:*

$$dist^{h,\phi}(p, p') = \frac{\sum_{v \in \mathcal{V}_{Ann}} \texttt{VAL-FUNC}(v, v^{h,\phi}, p, p')}{\mid \mathcal{V}_{Ann} \mid}$$

where `VAL-FUNC` is some function measuring a property of the effect of the valuation over the two polynomials.

`VAL-FUNC` functions. We next give examples for natural choices of $\texttt{VAL-FUNC}(v, v', p, p')$. In all examples $w(v)$ is some weighting over the valuation, e.g. the joint probability of the truth values it defines.
(1) **Expected error**: $w(v) \cdot \mid v(p) - v'(p') \mid$. Using $w(v) = 1$ leads to comparing the overall error over all truth valuations out of the given set. This scenario makes sense when provenance is likely to be performed for multiple valuations, and it is all right to suffer some small error in each computation.
(2) **Weighted fraction of disagreeing valuations**: 0 if $v(p) = v'(p')$ and w(v) otherwise. Using $w(v) = 1$ would be a reasonable choice if the user is to uniformly sample valuations and is interested in the probability of obtaining a correct/incorrect answer.
(3) **Euclidean distance**: $\texttt{euclidean-dist}(v(p), v(p'))$. This is well-defined when v(p) and v(p') are aggregation vectors rather than aggregation values (e.g. a vector of aggregated ratings of different movies of same genre).

EXAMPLE 3.4. *Observe that using $\mid v(p) - v'(p') \mid$ as the `VAL-FUNC`, P_s'' is at distance* 0 *from P_s w.r.t. valuations that map only a single user annotation to False. All these valuations yield the same value w.r.t. the two provenance expressions – if U_2 is mapped to True then the aggregated MAX value is* 5 *regardless of other truth values, and otherwise both U_1 and U_3 are mapped to True and so is Audience. In contrast, P_s' differs from P_s for the valuation that maps U_2 to False and the rest to True.*

Putting it all together. In addition to obtaining a provenance summary with small distance, we of course wish to minimize the provenance expression size. The distance and size measurements are combined together to form a weighted

average, where the weights are given as input parameters, that is used as a score given to candidate mappings. We later describe how this score is used in our summarization algorithm and also how taxonomy distances are incorporated in the computation.

DEFINITION 3.5. *Let p_0 be an input provenance expression and let $p_{cand} = h(p_0)$. Also, let wDist and wSize be user-defined weights ($wDist + wSize = 1$), rDist the approximated distance rank of p_{cand} and rSize its size rank. We define a candidate mapping score as follows:*
$CandidateScore^{h,\phi}(p_0, p_{cand}) =$
$wDist \times rDist^{h,\phi}(p_0, p_{cand}) + wSize \times rSize(p_{cand})$

Computational problems. Given a provenance expression p_0 with a set of annotations Ann, ϕ and `VAL-FUNC` functions, our goal is to explore the tradeoff between distance and size. This is studied in three flavors:
(1) using input weights for size and distance, for obtaining a homomorphic expression p', in each algorithm iteration, that minimizes the function CandidateScore.
(2) minimizing the distance while obtaining a summary p' of size less than some size bound `TARGET-SIZE`.
(3) minimizing the size while obtaining a summary p' of distance less than some distance bound `TARGET-DIST`.
These three flavors are all studied using the summarization algorithm. To use the first flavor the user can choose weights and bounds according to her preferences. For the second flavor, the user must set the distance weight to 1 and `TARGET-DIST` to the maximal distance (1). For the third flavor, the user must set the size weight to 1 and `TARGET-SIZE` to the minimal size (1). We study both a variant where the distance is computed with respect to all possible valuations (and then V_{Ann} is not an explicitly given input) as well as a variant where a subset V_{Ann} of valuations is given as input.

4. COMPUTING PROVENANCE SUMMARIZATIONS

There are two main building blocks in a solution that summarizes a provenance expression. The first is, given a summary, compute its quality based on the measurements discussed above. The second is a search algorithm that explores multiple possible expressions, uses the first building block to compute the quality for each, and aims at finding the best ones. We next detail the two components.

4.1 Computing Summary Quality

Recall that a summary quality was defined through the notion of distance. Let **DIST-COMP** be the problem of computing the exact distance (w.r.t. all possible valuations) between two provenance expressions p and p'=h(p), given input p, h, ϕ and `VAL-FUNC`.

PROPOSITION 4.1. ***DIST-COMP*** *is $\sharp$P-hard in the size of the input provenance p. This is true even if p includes no tensor elements.*

The proof (in [2]) is by reduction from $\sharp$-DNF. On the other hand, approximating the distance is feasible.

PROPOSITION 4.2. *Given a provenance expression p and a homomorphism h on its annotations, and given $0 < \epsilon, 0 < \delta < 1$, one can compute d' such that $Prob(\mid d' - dist(p, h(p)) \mid > \epsilon) < 1 - \delta$. The computation of d' may be performed in polynomial time with respect to $\mid p \mid$, δ, and $\frac{1}{\epsilon}$.*

The proof (in [2]) is constructive in the sense that it involves a simple sampling-based approximation algorithm, that will be used as a building block in our summarization algorithm.

4.2 Finding a Summarization

Towards a summarization algorithm, we recall that the set of truth valuations V_{Ann} may be restricted, guided by the intended use (in the sequel we will assume that V_{Ann} is given as input). In this case, we observe that V_{Ann} may already dictate some simplifications that may be performed (see below).

PROPOSITION 4.3. *Given a provenance expression p, finding a minimal p' such that $distance(p, p') = 0$ is in PTIME in p and in V_{Ann}.*

Equivalence Classes. The proof for the above proposition (in [2]) is based on computing equivalence classes of annotations with respect to a set of valuations, with every two annotations being equivalent if they agree for *every* valuation in the set. The intuition is that there is no need to maintain these different annotations, since they in any case may not be differentiated. Replacing them by ("mapping them to") the same annotation will further allow simplifications of the expression based on the algebraic identities. This calls for a first step in the summarization algorithm, grouping together provenance annotations that are equivalent. Of course, this may still yield expressions of large size; we will thus perform a A^*-like search ([26]) of expressions, motivated by the next proposition.

Input: A provenance expression p_0 with a set of annotations Ann, ϕ and `VAL-FUNC` functions, distance and size weights, size bound `TARGET-SIZE` and distance bound `TARGET-DIST`
Output: Summary provenance expression p_1

```
p' := GroupEquivalent(p_0, V_Ann) ;
while Size(p') > TARGET-SIZE or
ApproxDistance(p_0, p', V_Ann) < TARGET-DIST do
    for every h ∈ CandidateHom(p') do
        p_cand := h(p') ;
        if CandidateScore^{h,φ}(p_0, p_cand) is minimal
        then
            p'_prev := p' ;
            p' := p_cand ;
        end
    end
end
if ApproxDistance(p_0, p', V_Ann) ≥ TARGET-DIST then
    return p'_prev;
end
return p' ;
```

Algorithm 1: Provenance Summarization Algorithm

Monotonicity. Let $p_0, p_1, ..., p_n$ be provenance expressions such that $p_i = h_i(p_{i-1})$ for some sequence of homomorphisms h_i. We define monotonicity of the distance and

size functions, as follows: the distance function is increasing monotone iff for all $i > j$: $distance(p_0, p_i) \geq distance(p_0, p_j)$ and the size function is decreasing monotone iff for all $i > j$: $size(p_i) \leq size(p_j)$.

PROPOSITION 4.4. *All the* `VAL-FUNC` *functions described in section 3 yield increasing monotone distance and decreasing monotone size functions.*

Naturally, not every choice of `VAL-FUNC` leads to monotonicity, e.g. a function that returns alternating constants, but, as the above proposition indicates, natural choices of functions do.

Provenance Summarization Algorithm. The above proposition (proof in [2]) leads to Algorithm 1. Starting from the original set of annotations *Ann* and the given provenance expression p_0, the heuristic algorithm constructs the homomorphism h gradually, essentially by deciding on grouping of annotations. First, we obtain p' by grouping annotations that are equivalent w.r.t. the set of truth valuations (GroupEquivalent in line 1), as indicated by Proposition 4.3. Then, we iterate and in each step examine a set of possible single-step mappings (in $CandidateHom$) of two annotations to the same, new annotation name (line 3). For each such mapping we apply the obtained homomorphism to the current expression, computing $h(p')$ (line 4) and approximating the distance between p_0 and $p_{cand} = h(p')$. The p_{cand} with the smallest `CandidateScore` value is chosen (lines 5-8) and the process repeats until the stop condition is met. The stop condition for `TARGET-SIZE` (`TARGET-DIST`) is when the expression meets the size (resp. distance) bound (line 2).

If multiple candidates have minimal candidate scores, input taxonomies, if given as input, are used to break ties. For each such candidate, the taxonomy distances of the annotations from the new annotation they are mapped to are computed and the MAX (or SUM) of these distances is computed. The candidate that minimizes this value is chosen. If no taxonomies are given as input, we arbitrarily choose a candidate with minimal score.

EXAMPLE 4.5. *Returning to our running example, assume now that the Movies table also includes a movie genre column. Further assume that the user would like to view scores of movies of certain genres and so the aggregator now aggregates multiple movies of the same user-specified genre. We next exemplify the algorithm flow, using the following provenance expression for the movies "Match Point" and "Blue Jasmine":* $\boldsymbol{P_0} = P_{MP} \oplus_M P_{BJ}$ *where* $P_{MP} = P_s$ *is the provenance expression from Example 3.1 that consists of the three user reviews for the movie "Match Point",* $P_{BJ} = U_2 \otimes (4, 1)$ *is the added review for the movie "Blue Jasmine" and* $\oplus_M$ *is a formal sum for combining reviews of different movies (we will later see how this formal sum is used).*

In each step, the algorithm examines the set of possible mappings of two annotations to the same new annotation. The mappings $U_1, U_2 \rightarrow Female$ *and* $U_1, U_3 \rightarrow Audience$ *discussed in Example 3.1 are such possible single-step mappings that are examined by the algorithm. For simplicity, assume these are the only possible mappings. The algorithm computes the new provenance expressions that the candidate mappings yield:*
$\boldsymbol{P_0'} = P'_{MP} \oplus_M P'_{BJ} = Female \otimes (5, 2) \oplus U_3 \otimes (3, 1) \oplus_M Female \otimes (4, 1)$
$\boldsymbol{P_0''} = P''_{MP} \oplus_M P''_{BJ} = Audience \otimes (3, 2) \oplus U_2 \otimes (5, 1) \oplus_M U_2 \otimes (4, 1)$
Also, a candidate score for each such candidate is computed. Assuming $wDist = 1$ *and* $wSize = 0$*, the candidate that is chosen in each step in the one that minimizes the distance from the original provenance* P_0*.*

Assume we compute the distance w.r.t. the class of valuations that cancel a single annotation and the euclidean distance `VAL-FUNC`*. Keep in mind that evaluating a valuation on this kind of provenance, that consists of reviews for different movies, results in a vector of aggregated ratings where each coordinate holds the aggregated rating of a certain movie. In this setting,* P_0'' *is at distance* 0 *from* P_0 *while* P_0' *differs from* P_0 *for the valuation that cancels* U_2*. This is due to the fact that by canceling* U_2 *in* P_0 *we cancel the maximum rating for "Match Point" and the only rating for "Blue Jasmine". Canceling* U_2 *in* P_0' *does not have a similar effect since we use a disjunction of the truth values (of* U_1, U_2*) as our* ϕ *function, and so the new annotation "Female" is assigned the value true. Obviously, the euclidean distance between the aggregation vectors that are the result of evaluating this valuation on* P_0 *and* P_0' *is greater than zero and so the overall distance over all considered valuations is greater than zero. This leads to the conclusion that the algorithm would choose* P_0'' *over* P_0' *so the provenance for the next algorithm iteration is* $P_1 = P_0''$ *and so the algorithm continues.*

5. DATASETS AND USE CASES

We next describe the three provenance datasets that we used and show provenance examples for these datasets.

Datasets. We used three datasets: (1) MovieLens dataset, that includes ratings of different movies by users of MovieLens movie recommender that is based on collaborative filtering ([1]). (2) Wikipedia dataset - collected using the MediaWiki web API which is a Web service that provides convenient access to wiki features, data, and meta-data over HTTP. We also used YAGO Taxonomy ([5]) that contains rdfs:subClassOf facts derived from Wikipedia and WordNet. This taxonomy was used in our provenance summarization algorithm in order to improve the choices made by the algorithm when the input is a Wikipedia provenance expression. We used Wu-Palmer method for measuring semantic relatedness ([28]) in order to compute the distance between WordNet concepts in the taxonomy. (3) Data-Dependent Processes (DDP's) dataset - we generated provenance expressions that represent data-dependent processes based on the structure described in [17].

EXAMPLE 5.1. *A Wikipedia provenance expression represents different user edits of Wikipedia pages that belong to different categories. Each user edit can either be minor (0) or major (1). Consider the following provenance expression:*
$\boldsymbol{P_0} = (SalubriousToxin \cdot Adele) \otimes (0, 1) \oplus$
$(Dubulge \cdot CelineDion) \otimes (1, 1) \oplus$
$(Dr.\ Back\text{-}In\text{-}The\text{-}Street \cdot LoriBlack) \otimes (1, 1) \oplus$
$(JaspertheFriendlyPunk \cdot AlecBaillie) \otimes (1, 1).$
This provenance includes 3 major (Dubulge, Dr. Back-In-The-Street and Jasper the Friendly Punk) and 1 minor (Salubrious Toxin) user edits of 4 Wikipedia pages - 2 pages whose title is a famous singer (Adele and Celine Dion) and 2 whose title is a famous guitarist (Lori Black and Alec Baillie).

To simplify the example, assume we only map user annotations to the same annotation if the associated users have a similar number of edits and then the new annotation would describe their contribution level, e.g. "Top-Contributor" and "Reviewer". Also assume we map Wikipedia pages to a new annotation only if the corresponding pages have the same parent WordNet concept in the taxonomy. Moreover, assume we use SUM aggregation and that we compute the distance w.r.t. $\phi = \vee$ and the class of valuations that cancel a single annotation and are consistent with the taxonomy. A valuation is considered to be inconsistent if it assigns false to a Wikipedia category/WordNet concept A, but assigns true to a Wikipedia category/WordNet concept B s.t. B is a child of A in the taxonomy. A summarization is:

$\boldsymbol{P'} = (Top\text{-}Contributor \cdot <wordnet\_guitarist>) \otimes (2,2) \oplus (Reviewer \cdot <wordnet\_singer>) \otimes (1,2)$.
According to the summary, two users that are top contributors edited Wikipedia pages of guitarists (one major edit each) and two simple reviewers edited Wikipedia pages of singers (one major edit and the other minor).

By obtaining such a provenance summary it is easier to answer questions such as: what are the most controversial or interesting topics, what are relatively popular topics among top contributors, do top contributors make more major edits relative to other users, etc (e.g. obtaining a summary similar to the above summary for many Wikipedia users might lead us to the conclusion that top contributors prefer to edit guitarist pages than singer pages). These are questions that are much harder to answer using the original long provenance expression. We can also present such summaries in a ui that makes it easier for the user to understand the summary and get insights on the underlying data.

EXAMPLE 5.2. *A DDP (Data Dependent Process), described in [17], models an application whose control flow is guided by a finite state machine, as well as by the state of an underlying database. DDP provenance expressions are summaries of executions where an execution is a multiplication of transitions. Each transition is either based on a user's choice or on a database query result. A user dependent transition is of the form $\langle c_k, 1\rangle$ where c_k is the cost associated with the transition (the user's effort). A database dependent transition is of the form $\langle 0, [d_i \cdot d_j] \neq 0\rangle$ or $\langle 0, [d_i \cdot d_j] = 0\rangle$. Consider the following DDP provenance example of two executions (each consisting of two transitions):*
$\langle c_1, 1\rangle \cdot \langle 0, [d_1 \cdot d_2] \neq 0\rangle + \langle 0, [d_2 \cdot d_3] = 0\rangle \cdot \langle c_2, 1\rangle$.

The aggregation function used is based on the semirings described in [17]. The "attributes" that are used as constraints here are the mappings of different database variables to new database variables and the mappings of cost variables to new cost variables. If two database variables are mapped to a single one, it means that either both tuples need to be present for the database query to be satisfied or both should be missing. Similarily, if we know that user transitions have more or less the same cost, it is possible to map the two cost variables to a new cost variable. Also, assume we use the "Cancel Single Attribute" class of valuations.

The following is a possible summary for the above provenance, obtained by mapping d_1, d_3 to D_1 and c_1, c_2 to C_1:
$\langle C_1, 1\rangle \cdot \langle 0, [D_1 \cdot d_2] \neq 0\rangle + \langle 0, [d_2 \cdot D_1] \neq 0\rangle \cdot \langle C_1, 1\rangle$
which is equal to: $\langle C_1, 1\rangle \cdot \langle 0, [D_1 \cdot d_2] \neq 0\rangle$.
This final summary represents a single execution that consists of two transitions - one user dependent transition and one database dependent transition.

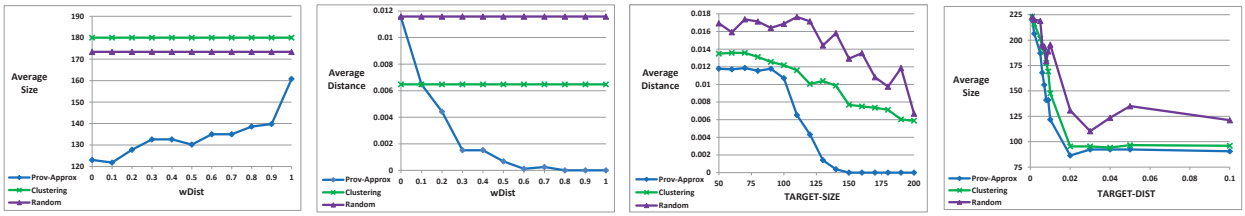
By summarizing this kind of provenance, analysts can test and explore the effect of hypothetical modifications to a DDP's state machine and/or to the underlying database (e.g. using the above summary, analysts can explore the effect of removing the database dependent transition). Exploring the effect of such modifications using the original provenance expression can be much more complicated.

6. EXPERIMENTS

The main purpose of our experiments was to examine the effectiveness of our summarization algorithm, compared to other approaches, in terms of: (1) conciseness of the obtained provenance expression (measured by size), (2) accuracy of evaluations (measured by distance from original provenance), (3) faster provenance usage ("Usage Time" experiment) and (4) feasibility of summarization ("Summarization Time" experiment). The first two were examined as functions of the wDist weight (wDist experiment), and of the `TARGET-DIST` and `TARGET-SIZE` stop conditions (`TARGET-DIST` and `TARGET-SIZE` experiments). This covers the three computational problems that we have presented in section 3. Each experiment was conducted for the three datasets - MovieLens, Wikipedia and DDP. For each dataset, we generated multiple input provenance expressions, executed the experiments and averaged the results.

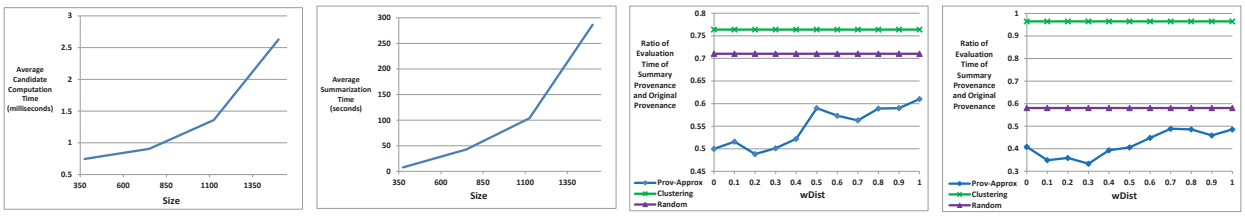
Algorithms Examined. In each experiment that we conducted, we executed the following algorithms for each dataset and compared different parameters of the result summary provenance: (1) Prov-Approx (Algorithm 1) - our provenance summarization algorithm. (2) Clustering - using only the MovieLens and Wikipedia datasets, since the DDP dataset is not suitable for the clustering experiment (it is not clear how to construct feature vectors to be used as input to the clustering algorithm). (3) Random - in which every pair of annotations was chosen randomly from the list of pairs that satisfy the mapping constraints. All three algorithms take into account the user-specified size and distance bounds (`TARGET-SIZE` and `TARGET-DIST`) and stop if and when they reach these bounds.

Clustering Algorithm. We used a library for hierarchical agglomerative clustering called HAC ([4]). This library supports the following linkage criteria (i.e. a criteria that determines the distance between sets of observations as a function of the pairwise distances): Single Linkage, Average Linkage, Centroid Linkage, Complete Linkage, Median Linkage, Ward Linkage and Weighted Average, described in [13]. All linkage criteria were examined in the experiments, but since they all yield similar results compared to our approach we present the "Single Linkage" results. Next we describe how the clustering approach was implemented for the MovieLens dataset. Similarly, it was also implemented for the Wikipedia dataset. Not all provenance datasets are suitable for a Clustering algorithm, e.g. our DDP dataset. Each user, that rated k movies, was associated with a feature vector of the following form:
$(UID, Gender, AgeRange, Occupation, ZipCode,$
$(MovieTitle_1 = Rating_1, ..., MovieTitle_k = Rating_k))$, e.g.
$(UID278, M, 45-49, tradesman/craftsman, 60482,$



(a) Size-wDist (b) Dist-wDist (c) Dist-TARGET-SIZE (d) Size-TARGET-DIST

Figure 2: Average Distance as a Function of wDist and `TARGET-SIZE` and Average Size as a Function of wDist and `TARGET-DIST` (MovieLens Dataset)



(a) Time Per Candidate (b) Summarization Time (c) Ratio of Usage Time of Summary Provenance and Original Provenance (Using 20 Steps) (d) Ratio of Usage Time of Summary Provenance and Original Provenance (Using 30 steps)

Figure 3: Summarization and Candidate Computation Time (MovieLens Dataset)

$(TheFury = 4.0, NearDark = 4.0)$).
In addition, we implemented a dissimilarity measure for computing the distance between each pair of feature vectors. We used Pearson Correlation Coefficient as a measure of similarity between the ratings vectors, that the feature vectors include as a single feature. Moreover, we added our mapping constraints to the clustering algorithm so that both our algorithm and the clustering algorithm would take into account the same constraints (we do not allow two clusters to merge if the users that belong to these clusters do not have at least one attribute in common).

We next describe how we obtain the Clustering's provenance summary in order to compare its quality to ours. Similarily to our summarization algorithm, each step of the Clustering algorithm, in which two clusters are merged, corresponds to a mapping of 2 annotations to an annotation summary. According to this mapping, we compute the obtained provenance expression and use it to check the stop conditions - `TARGET-DIST`, `TARGET-SIZE`, etc.

Experimental Settings. The experiments were conducted for different combinations of datasets, valuation classes and aggregation functions and all combintaions have similar results. Specifically, two valuation classes were examined: (1) "Cancel Single Annotation" - each valuation in this class cancels a single annotation by assigning it $false$ and assigning $true$ to the rest. (2) "Cancel Single Attribute" - the class of valuations that cancel all annotations that share the same attribute and assigns $true$ to the rest (e.g. the valuation that cancels all Male users). For space constraints, we present a set of representative results. It is important to note that the distance values we present, represent average error over all valuations, which we divide by the maximum possible error in order to normalize to [0,1]. Presenting the un-normalized distances results in the same graph trends.

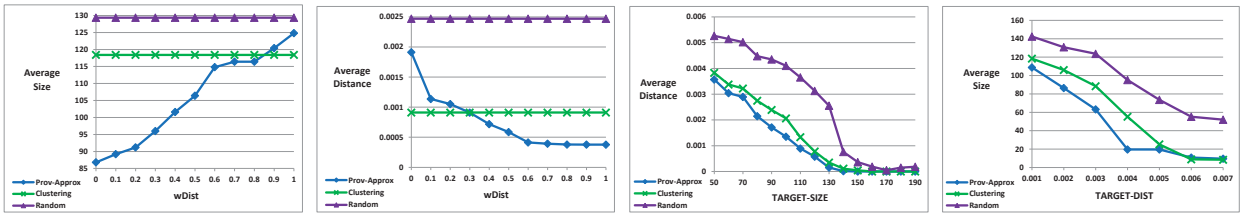
We next describe our experimental results for the MovieLens dataset. Later we show some experimental results for the other two datasets.

6.1 wDist Experiment

The purpose of this experiment is to check the effect of the wDist and wSize weights on the summary distance and size. For that purpose, we bounded the maximum number of algorithm steps, the `TARGET-DIST` was set to 1 (max distance) and the `TARGET-SIZE` was set to 1 also (minimum size) so that they would have no effect as stop conditions. Figures 2a and 2b show the results for the MovieLens dataset using "Cancel Single Attribute" valuation class, MAX aggregation and at most 20 steps. As expected, using Prov-Approx, greater values of wDist yield smaller distance values and greater size values. The wDist has no effect on the Clustering and Random approaches (they do not take this parameter into account) so we averaged their results for different wDist values. As the wDist used increases, Prov-Approx yields smaller distance compared to the Clustering (starting from wDist = 0.1 as presented in the graph), as expected. Also, the Clustering approach yields greater size compared to our approach. The Random approach yields much greater distance and size values.

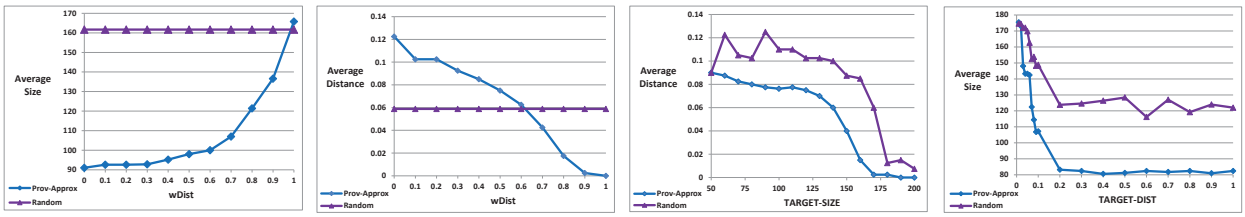
6.2 TARGET-SIZE Experiment

This experiment checks the problem variant in which the user aims to reach a certain `TARGET-SIZE` value while keeping the result relatively "close" to the original provenance. For that purpose, we set the wDist and `TARGET-DIST` values to 1. Figure 2c shows the results of the experiment for the MovieLens dataset. As expected, since the wDist was set to 1, our approach gave better distance values compared to the Clustering and Random approaches. The Random approach gave the worst results. For the Prov-Approx and the Clustering algorithms, as the `TARGET-SIZE` increases, the



(a) Size-wDist (b) Dist-wDist (c) Dist-TARGET-SIZE (d) Size-TARGET-DIST

Figure 4: Average Distance as a Function of wDist and `TARGET-SIZE` and Average Size as a Function of wDist and `TARGET-DIST` (Wikipedia Dataset)



(a) Size-wDist (b) Dist-wDist (c) Dist-TARGET-SIZE (d) Size-TARGET-DIST

Figure 5: Average Distance as a Function of wDist and `TARGET-SIZE` and Average Size as a Function of wDist and `TARGET-DIST` (DDP Dataset)

size increases (the algorithms stop earlier) and as a result the distance is smaller. This is not always the case for the Random approach. Since this approach does not make the same choices for all the `TARGET-SIZE` values but just randomly chooses a pair, there could be better distance choices made when smaller `TARGET-SIZE` values are used (e.g. the distance that Random achieved using `TARGET-SIZE` 130 is actually smaller than the one achieved using 140).

6.3 TARGET-DIST Experiment

This experiment checks another variant, in which the user wishes to bound the distance with the `TARGET-DIST` value while obtaining a small provenance expression. For that reason, wDist was set to 0 and `TARGET-SIZE` was set to 1 also. Figure 2d shows the results of the experiment for the MovieLens dataset. As expected, as the `TARGET-DIST` increases the size decreases until we reach a point from where we cannot decrease the size further. Moreover, as a result of using wDist = 0 the choices made were the ones that minimize the next expression's size and that reflects in the results, where our approach reaches the smallest size values. Random gives the worst results. Also, Random does not make the same choices for all `TARGET-DIST` values, so it is also possible that it would make better choices for smaller `TARGET-DIST` values and that would yield better size, just like in `TARGET-DIST` 0.03.

6.4 Usage Time Experiment

This experiment examines the ratio between the average evaluation time of valuations on the summary and original expressions, as a function of wDist. Figures 3c and 3d show the results for the MovieLens dataset using 20 and 30 maximum number of steps respectively. The `TARGET-DIST` was set to 1 and the `TARGET-SIZE` to 1 so that the only relevant stop condition would be the number of steps. The experiment was conducted as follows: we randomly chose 10 valuations, evaluated these valuations on the original and summary expressions for the three approaches and examined the ratio of evaluation time. As expected, using Prov-Approx, as the wDist increases, the result's size is greater and the distance is smaller. For that reason, the expression is closer to the original expression and so the ratio in evaluation time is greater. Also, using more algorithm steps, the ratio is smaller; using 30 steps the range is 0.3-0.5 (30% - 50% improvement in evaluation time) compared to 0.45-0.65, when using 20 steps. To conclude, the summary usage time is faster than the original provenance usage time. In addition, the Random and Clustering approaches are not affected by wDist, so we averaged the results for all the wDist values. As expected, our approach yields smaller ratio compared to the Random approach using smaller wDist values. The Clustering approach yields much greater ratio than ours for all wDist values (less improvement in usage time).

6.5 Summarization and Candidate Computation Time Experiment

This experiment examines the summarization time and also the average candidate computation time (distance and size computation for a candidate pair of annotations) as functions of provenance size. Figures 3a and 3b show the results for the MovieLens dataset, wDist weight set to 1 and 50 maximum number of steps. As expected, as the expression size decreases, the number of pairs to consider in each step decreases and so the number of distance calculations decreases and as a result the summarization time decreases. Also, as the expression size decreases, the distance computation is faster, as expected.

6.6 Other Datasets

All the figures so far show results for the MovieLens dataset; we next describe results for the other two datasets. Figures 4a, 4b, 5a and 5b show the results of the wDist experiment conducted on the Wikipedia and DDP datasets using 20 and 10 maximum number of steps respectively. Figures 4c, 4d, 5c and 5d show the results of the `TARGET-SIZE` and `TARGET-DIST` experiments for these datasets, using "Cancel

Single Annotation" valuation class and sum aggregation for the Wikipedia dataset and "Cancel Single Attribute" for the DDP dataset. All results are similar to those obtained for the MovieLens dataset. Note that the DDP dataset is the only one that wasn't compared to the Clustering approach since it's unclear how to construct a Clustering competitor for this complex-structured data provenance.

7. RELATED WORK AND CONCLUSIONS

Provenance models have been extensively studied in multiple lines of research such as provenance for database transformations (see e.g. [18, 11, 9, 21, 10, 12]), for workflows (see e.g. [16, 14, 6, 22, 19, 27, 23]), for the web [3], for data mining applications [20], and many others, but typically full and exact provenance is presented (sometimes in an optimized form, e.g. factorized as in [24]). Provenance views have been proposed in context of workflows (see e.g.[16]), but the summarization obtained through these views is based on a notion of granularity levels, and is lossless rather than approximate. A notion of approximate provenance was proposed in [25], and somewhat resembles ours, but is limited to UCQs (and in particular allows no aggregates), geared towards probabilistic computation, and does not account for semantic constraints. Our notion of mapping to summarized annotations is also reminiscent of clustering, however the function that we optimize is one that depends on the provenance expression itself and its intended uses, which leads to different design choices and to different results.

We have studied in this paper summarization of provenance information. We have identified three desiderata for the assessment of candidate summaries: conciseness, semantic constraints satisfaction and small distance from original provenance. This has led us to the development of our summarization algorithm that finds an "optimal" summary according to these quality measurements. After comparing our approach to other approaches (Clustering and Random) by conducting different experiments using different provenance datasets, we conclude that our approach is indeed better - it finds better quality summaries compared to the others and allows the user to control the desired tradeoff between distance (that affects evaluation accuracy) and size (that affects presentation and usage time). As future work, we intend to explore a generalized version of the algorithm in which in each iteration we map k annotations to a new annotation rather than just 2. An additional line of future work is to acheive further theoretical bounds on the algorithm's performance and output quality.

8. ACKNOWLEDGMENTS

This work has been partially funded by the European Research Council under the FP7, ERC grant MoDaS, agreement 291071, by the Broadcom Foundation and Tel Aviv University Authentication Initiative, by the Israeli Science Foundation (Grant No. 1636/13), by the National Science Foundation Institute (NSF), Information and Intelligent Systems (IIS) Division (Grant No. 1302212) and by the French National Research Agency (ANR), Aggreg project.

9. REFERENCES

[1] Movielens site. `https://movielens.org/`.
[2] Paper's online version. `http://bit.ly/1ceBKEC`.
[3] Provenance working group. `http://www.w3.org/2011/prov/`.
[4] Sape research group. `http://sape.inf.usi.ch/hac`.
[5] Yago knowledge base. `http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/`.
[6] A. Ailamaki, Y. E. Ioannidis, and M. Livny. Scientific workflow management by database management. In *SSDBM*, 1998.
[7] Y. Amsterdamer, S. B. Davidson, D. Deutch, T. Milo, J. Stoyanovich, and V. Tannen. Putting Lipstick on Pig: Enabling Database-style Workflow Provenance. *PVLDB*, 2012.
[8] Y. Amsterdamer, D. Deutch, and V. Tannen. Provenance for aggregate queries. In *PODS*, 2011.
[9] O. Benjelloun, A. Sarma, A. Halevy, M. Theobald, and J. Widom. Databases with uncertainty and lineage. *VLDB J.*, 17, 2008.
[10] P. Buneman, J. Cheney, and S. Vansummeren. On the expressiveness of implicit provenance in query and update languages. *ACM Trans. Database Syst.*, 33(4), 2008.
[11] P. Buneman, S. Khanna, and W. Tan. Why and where: A characterization of data provenance. In *ICDT*, 2001.
[12] J. Cheney, L. Chiticariu, and W. C. Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474, 2009.
[13] P. R. Christopher D. Manning and H. SchÃijtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
[14] D. Cohn and R. Hull. Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.*, 32(3), 2009.
[15] S. B. Davidson, S. C. Boulakia, A. Eyal, B. Ludäscher, T. M. McPhillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4):44–50, 2007.
[16] S. B. Davidson and J. Freire. Provenance and scientific workflows: challenges and opportunities. In *SIGMOD Conference*, pages 1345–1350, 2008.
[17] D. Deutch, Y. Moskovitch, and V. Tannen. A provenance framework for data-dependent process analysis. *PVLDB*, 7(6):457–468, 2014.
[18] R. Fink, L. Han, and D. Olteanu. Aggregation in probabilistic databases via knowledge compilation. *PVLDB*, 5(5), 2012.
[19] I. Foster, J. Vockler, M. Wilde, and A. Zhao. Chimera: A virtual data system for representing, querying, and automating data derivation. *SSDBM*, 2002.
[20] B. Glavic, J. Siddique, P. Andritsos, and R. J. Miller. Provenance for Data Mining. In *Theory and Practice of Provenance (TAPP)*, 2013.
[21] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40, 2007.
[22] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. Pocock, P. Li, and T. Oinn. Taverna: a tool for building and running workflows of services. *Nucleic Acids Res.*, 34, 2006.
[23] P. Missier, N. Paton, and K. Belhajjame. Fine-grained and efficient lineage querying of collection-based workflow provenance. In *EDBT*, 2010.
[24] D. Olteanu and J. Zavodny. Factorised representations of query results: size bounds and readability. In *ICDT*, pages 285–298, 2012.
[25] C. Ré and D. Suciu. Approximate lineage for probabilistic databases. *PVLDB*, 1(1):797–808, 2008.
[26] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 2009.
[27] Y. L. Simhan, B. Plale, and D. Gammon. Karma2: Provenance management for data-driven workflows. *Int. J. Web Service Res.*, 5(2), 2008.
[28] Z. Wu and M. Palmer. Verb semantics and lexical selection. In *Proc. of 32nd Annual meeting of the Associations for Computational Linguistics*, pages 133–138, 1994.