A Model for Fine-Grained Data Citation*

Susan Davidson[1], Daniel Deutch[2], Tova Milo[2], and Gianmaria Silvello[3]

[1] University of Pennsylvania, USA
susan@seas.upenn.edu
[2] Tel Aviv University, Israel
{danielde,milo}@post.tau.ac.il
[3] University of Padua, Italy
silvello@dei.unipd.it

Abstract. An increasing amount of information is being collected in structured, evolving, curated databases, driving the question of how information extracted from such datasets via queries should be cited. Unlike traditional research products which have a fixed granularity, data citation is a challenge because the granularity varies. Different portions of the database, with varying granularity, may have different citations. Furthermore, there are an infinite number of queries over a database, so we cannot hope to explicitly attach a citation to every possible result set and/or query. We present the novel problem of *automatically generating citations for general queries* over a relational database, and explore a solution based on *citation views*, each of which attaches a citation to a view of the database. Citation views are then used to automatically construct citations for general queries.

1 Introduction

Citation is essential to traditional scholarship. It helps identify the cited material so that it can be retrieved, gives credit to the creator of the material, dates it, and so on. In the context of printed materials, such as books and journals, citation is well understood. However, the world is now digital, and an increasing amount of information is being collected in structured and evolving curated databases, driving database owners, publishers and standards groups to consider how such data should be cited [6].

Unlike traditional publications which have a fixed granularity to which citations can be attached – e.g. a conference proceedings, or a paper in a conference proceedings – data citation is a challenge because the granularity varies. For example, the content of a scientific database frequently represents the effort of a large number of members of the scientific community (some use cases are presented in Section 2). Different portions of the database, with varying granularity, are contributed and/or curated by different subgroups of these individuals. While a citation to the database as a whole is typically provided as a traditional publication, whose author list includes the owners and developers of

* This is an extended abstract of [5]. Please refer to the original paper for more details.

the resource, there is a growing belief that contributors/curators should be acknowledged when their data is extracted and used. That is, when the database is queried and a result set returned, the citation (i.e. a textual snippet to be used as a reference) for that data should include information about the contributors/curators of the result set as well as of the data used to compute it. The latter depends on the query.

Since there are a potentially infinite number of queries, each accessing and generating different subsets of the database, we cannot hope to explicitly attach a citation to every possible result set and/or query. Instead, we must find ways of specifying citations for some semantically meaningful portions of the database (possibly defined declaratively via queries), and use these to automatically construct citations for more general queries. Thus data citation is a *computational* problem, as argued in [3] and fleshed out in more detail here.

Our approach draws inspiration from results in two areas: *query rewriting using views*, which has been well studied in the context of query optimization, data integration [2], and *database provenance* [7].

We start with a notion of *citation views*, through which database owners specify how citations are attached to the output of some number of queries (views) over the database representing typical usage patterns. The citation to a general query is then constructed by *rewriting* it to a set of equivalent queries using the citation views, and combining the citations attached to these views to construct a citation to the query (Section 3). To construct the citation, we leverage the fact that citations and provenance are both forms of *annotation* that are manipulated through queries [4]. In particular, the *joint* ($\cdot$) and *alternative* ($+$) use of annotations within a rewriting are modeled using the *semirings* approach of [7] presented in Section 4. We close in Section 5 by discussing several open issues that must be addressed in developing a practical solution to this challenging problem.

2 Use cases

Currently, several data resources specify *how* citations to single resources, collections, result lists and data subsets should be constructed but do not generate the appropriate citation. Three such examples are Eagle-i[4] (an RDF dataset built to facilitate translational science research which allows researchers to share information about resources such as cell lines and software), Reactome[5], an open-source, curated and peer reviewed pathway relational database, and Drugbank[6], a relational database (available also as RDF and XML) combining detailed chemical, pharmacological and pharmaceutical data with comprehensive sequence, structure, and pathway information. Instructions are given on which snippets of information on the web page view of the resource should be included in a citation to the resource, but generating the citation is left to the user.

[4] `https://www.eagle-i.net/`
[5] `http://www.reactome.org/`
[6] `http://www.drugbank.ca/`

The example we will use throughout the remainder of this paper is the IUPHAR/BPS Guide to Pharmacology (GtoPdb).[7] GtoPdb is a relational database that contains expertly curated information about drugs in clinical use and some experimental drugs. Users view information through a hierarchy of web pages: The top level divides information by families of drug targets that reflect typical pharmacological thinking; lower levels divide the families into subfamilies and so on down to individual drugs. The content of a particular family "landing" page is curated by a committee of experts; a family may also have a "detailed introduction page" which is written by a set of contributors, who are not necessarily the same as the committee of experts for the family. Currently, citations for these views are hard-coded into the web pages – queries are issued against the underlying relational database to obtain content for the page as well as the committee members or contributors. Thus, GtoPdb in fact does generates citations, but only to a subset of the possible queries against the underlying relational database, i.e. those corresponding to web-page views of the data.

In the future, the owners of GtoPdb would like to allow users to issue *general queries* against the relational database and automatically generate a citation for the result. We address this problem in the remainder of the paper.

3 Citation Views and Rewriting

We assume that queries and views are expressed as Conjunctive Queries (see [1] for an overview). Given a database D, we start with a set of *view definitions*. Each view definition has an associated *citation query* C_V and *citation function* F_V. A *citation view* consists of a view query, a citation query, and citation function. The citation function takes as input the citation query result, and returns a citation. The citation becomes an annotation on each tuple in the view query result. The view and citation queries may be *parameterized*, which we discuss shortly.

Definition 1. *A* citation view *is a triple (V, C_V, F_V) where*

- *V is the* view definition *of form $\lambda X.V(Y) :- Q$;*
- *C_V is the* citation query *of form $\lambda X.\ C_V(Y') :- Q'$; and*
- *F_V is the* citation function *which transforms the output of the citation query (i.e. the bindings to variables Y') into a citation in some desired format, such as JSON or XML.*

In V and C_V:

- *Q and Q' are conjunctions of atoms;*
- *$X = [x_1, ..., x_n]$, $Y = [y_1, ..., y_m]$, $Y' = [y'_1, ..., y'_p]$ are ordered sequences of variables;*
- *$X \subseteq Y$ are subsets of the variables in Q;*
- *X and Y' are subsets of the variables in Q'.*

[7] `http://www.guidetopharmacology.org/`

The terms $x_1, ..., x_n$ are the parameters *of V and C_V. Parameters are optional, in which case the λ-term may be dropped. If there is a λ-term the citation view is called* parameterized.

Each choice of values for the parameters may lead to a different view instance and different citation. We denote the view instance by applying the view to the input parameter values, e.g. if the view is

$$\lambda(x_1, \ldots, x_n)V(Y) :- Q$$

and is passed parameter values $(a_1, \ldots, a_n)$, we write

$$V(Y)(a_1, \ldots, a_n).$$

Since the citation query Q' is also parametrized by X, each valuation of X leads to a different result for Q'. The output of the citation function – the *citation* for the view – becomes an annotation on each tuple in the output of the instantiated view. Thus for every sequence of parameter values $(a_1, \ldots, a_n)$, the citation for all tuples in $V(Y)(a_1, \ldots, a_n)$ is $F_v(C_v(Y')(a_1, \ldots, a_n))$. Note that the citation views and queries have to be defined by the DBA and thus they can be considered a user choice. On the other hand, we can also foresee some methods to automatic define views and queries for instance by using query logs.

Example 1. We use as a running example a simplified database schema for GtoPdb, in which keys are underlined:

Family(<u>FID</u>, FName, Type)
FamilyIntro(<u>FID</u>, Text)
Person(<u>PID</u>, PName, Affiliation)
FC(<u>FID</u>, <u>PID</u>), FID references Family, PID references Person
FIC (<u>FID</u>, <u>PID</u>), FID references FamilyIntro, PID references Person
MetaData(Type, Value)

Intuitively, FC captures the committee members who curate the content of a family page while FIC captures the contributors who author the Family Introduction page of a family. The last table, MetaData, captures other information that may be useful to include in citations, such as the owner of the database ("Owner", "Tony Harmar"), the URL of the database ("URL", "guidetopharmacology.org") and the current version number of the database ("Version", "23").

To express citation views for GtoPdb we start by defining a small set of view definitions.

$$\begin{aligned} \lambda F.V1(F, N, Ty) &:- Family(F, N, Ty) \\ \lambda F.V2(F, Tx) &:- FamilyIntro(F, Tx) \\ V3(F, N, Ty) &:- Family(F, N, Ty) \end{aligned}$$

All views except $V3$ are parameterized. $V1$ and $V2$ restrict the output to a single tuple since the parameter, F, corresponds to the key FID in Family. $V3$ contains all tuples in Family. For each of the views, we define a citation query.

$$\begin{aligned} \lambda F.\ C_{V1}(F, N, Pn) &:- Family(F, N, Ty), FC(F, C), Person(C, Pn, A) \\ \lambda F.\ C_{V2}(F, N, Tx, Pn) &:- Family(F, N, Ty), FamilyIntro(F, Tx), \end{aligned}$$

$$C_{V3}(X1, X2) \quad :- \begin{array}{l} FIC(F, C), Person(C, Pn, A) \\ MetaData(T1, X1), T1 = \text{“}Owner\text{”}, \\ MetaData(T2, X2), T2 = \text{“}URL\text{”} \end{array}$$

Since V1 and its citation query C_{V1} are parameterized by F, each valuation of V1 results in a single tuple and each tuple may have a different citation. As an example, for the family with FID=11 the citation is a function of the result of C_{V1}(F,N,Pn)(“11”), which consists of the family id, name and list of committee members. In contrast to $V1$ and $V2$, $V3$ is not parameterized, which means that all tuples have the same citation in V3. C_{V3} uses the MetaData table to obtain the url and owner of the database.

Database owners specify a set of citation views, from which the citation for a general query over the database will be constructed. Our approach is to rewrite as much of the query as possible using the view definitions, and combine their citations to construct a citation for the input query. We start by defining what a rewriting is.

Definition 2. *Let $\mathcal{R}$ be a set of relation names, Q be a query, and $\mathcal{V}$ be a set of views. The query Q' is a* rewriting *of Q using $\mathcal{V}$ if:*

- *the subgoals of Q' are either relation names in $\mathcal{R}$, views in $\mathcal{V}$, or comparison predicates;*
- *Q' is equivalent to Q;*
- *no subgoal of Q' can be removed and obtain an equivalent query; and*
- *no subset of subgoals of Q' can be replaced by a view in $\mathcal{V}$ and obtain an equivalent query.*

A rewriting is *total* if its subgoals contain only views and comparison predicates; otherwise, if its subgoals also contain relation names, it is *partial*. We illustrate the trade-offs between different rewritings in terms of the citations generated through an example.

Example 2. Let us consider a query which finds the name and text of the introduction of families with type “gpcr”.

$$Q(N, Tx) :- Family(F, N, Ty), FamilyIntro(F, Tx), Ty = \text{“}gpcr\text{”}$$

Q can be rewritten in several ways, including:

$$Q_1(N, Tx) :- V1(F, N, Ty), V2(F, Tx), Ty = \text{“}gpcr\text{”}$$
$$Q_2(N, Tx) :- V3(F, N, Ty), V2(F, Tx), Ty = \text{“}gpcr\text{”}$$

These two rewritings are total and the difference between Q_1 and Q_2 is that the former uses V_1, while the latter uses V_3; C_{V1} produces a citation for each distinct family, whereas C_{V3} provides a single citation for the entire view. In this sense, $V3$ is more general than $V1$. In neither case is the comparison predicate $Ty =$ “$gpcr$” matched by the lambda term of the view.

4 Combining citations

As we have seen, a single query may be rewritten in multiple ways and each rewriting may use one or more views. The question is how the citation for the query result should be defined as a function of these rewritings. To do this, observe that citations and provenance are both forms of *annotation* that are manipulated through queries [4]. We therefore take inspiration from work on database provenance, in particular that of *provenance semirings* [7], to model the different ways in which citations views are combined.

In provenance semirings, provenance tokens (base annotations) are associated with each tuple in a relational instance (EDB). Restricting our attention to SPJU queries, there are two ways in which tuples are combined through queries:

- *Joint use*, as in a join which combines two tuples to form a new tuple. In this case, the provenance annotation of the new tuple is the '$\cdot$' of the annotations of the two input tuples.
- *Alternate use*, as when one or more tuples are "identified" via unions or projections to form a new tuple. In this case, the provenance annotation of the new tuple is the '+' of the annotations of all input tuples.

In citations, annotations are defined through the citation queries (and their corresponding functions). Citation views are combined in two different ways when providing a citation to general queries: they may appear *jointly* in equivalent rewritings of the query, or they may appear in *alternate* rewritings.

In the following, we will assign query output with citations that are the combination of results of citation functions, through + and $\cdot$ based on the use of the views in the query. The resulting structure of citations is that of a *commutative semiring*: we start with a set of basic citations C, and introduce an abstract operation + on it with the properties that + is commutative, associative, and has some neutral element 0 in C. Similarly we introduce an operation $\cdot$ with the same properties, but with a different neutral element 1. Last, we impose that $\cdot$ is distributive over +.

We start by defining a citation for a *single binding* of a *single rewriting* of the query. This dictates a single output tuple, and a particular valuation to the parameters of the views. We define the citation of the output tuples as the joint use of citations for the views and the parameter valuations, denoted by "$\cdot$".

Definition 3. *Let Q be a query, let V be a set of citation views and let Q' be a (partial) rewriting of Q using $V1, ..., Vn \in V$. Further let B be a binding to the variables of Q', yielding an output tuple t. The citation for t w.r.t. Q, Q', V, B, denoted as $cite(t, Q, Q', V, B)$, is defined as $F_{V1}(C_{V1}(B1)) \cdot \ldots \cdot F_{Vn}(C_{Vn}(Bn))$ where Bi is the application of B to the variables occurring in an atom involving Vi in Q'.*

Example 3. Consider the rewriting Q_1 from Example 2, and consider the binding to its variables F="11", N="Calcitonin", Ty = "gpcr" and Tx="stuff". The resulting tuple is ("Calcitonin", "stuff"), and the citation we get *for this particular binding* is the citation assigned in $V1$ to family "11" (note that the lambda

parameter of $V1$ is F), combined via $\cdot$ with the one assigned in $V2$ to the same family (i.e., $F_{V1} \cdot F_{V2}$): {ID: "11", Name: "Calcitonin", Committee: ["Hay", "Poyner"]} $\cdot$ {ID: "11", Name: "Calcitonin", Text: "The calcitonin peptide family ", Contributors: ["Brown", "Smith"]}.

So far we have only defined the citation for a single binding. Multiple bindings lead to multiple *alternative* citations, which we capture using $+$.

Definition 4. *Let Q, V, Q' be as in Definition 3, and let β_t be the set of all bindings for Q' that yield a tuple t. The citation for t w.r.t. Q, Q' (denoted as $cite(t, Q, Q', V)$) is denoted as $\Sigma_{B \in \beta_t} cite(t, Q, Q', V, B)$.*

Example 4. Recall $Q1$ and assume now that the family name N="Calcitonin" is shared by two families, with identifiers 11 and 12. This leads to two bindings to the variables of $Q1$, and intuitively to two ways of using the views to get the output tuple ("Calcitonin"). The citation for the tuple, in this case, will be a "+" over the expression in Example 3 and a similar expression for family id 12: {ID: "11", Name: "Calcitonin", Committee: ["Hay", "Poyner"]} $\cdot$ {ID: "11", Name: "Calcitonin", Text: "The calcitonin peptide family", Contributors: ["Brown", "Smith"]} + {ID: "12", Name: "Calcitonin", Text: "The calcitonin peptide family", Contributors: ["Brown", "Hey"]}

For example, "$\cdot$" could be a join of some sort and "+" could be union.

Finally, a query may have multiple rewritings, each leading to a possibly different citation. These are again alternatives, but the function used to combine the citations for them may be different than the one used for multiple bindings for a single rewriting. We therefore use $+^R$ ("+ for rewritings") to denote this function, whose operands are elements of the citation semiring (i.e. polynomials using $+$ and $\cdot$). $+^R$ has a neutral value 0_R, and is associative and commutative. Note that this is a formal semantics, not a means of computation: going through all rewritings would be an impractical implementation.

There are many interpretations "$\cdot$" , "+", and $+^R$ that could be used. For "$\cdot$" and "+", union or join are natural interpretations. For $+^R$, the "minimum" in some ordering would also be natural. These ordering could reflect how "precise" or "comprehensive" the rewritings are relative to each other, or the "size" of the resulting citation.

5 Final Remarks

We presented an approach for automatically constructing citations to information extracted from a database via general queries. In this approach, owners of the database specify citations to a small set of (possibly parameterized) views of the database which represent typical usage patterns; they also give interpretations to the combining functions $+$, $\cdot$ and $+^R$. A query against the database is then rewritten using the views, and a citation for the result set constructed using the interpretations of the combining functions.

The described model can be used for constructing citations using only schema level information (as is done in query optimization using views) as well as using citation annotations attached to tuples (as is done with reasoning about provenance). Note that reasoning at the tuple level requires changes to an existing database both in terms of the schema (to capture citation annotations on view tuples) and in terms of query processing (to combine citation annotations).

While we have formally defined a model for citations for query results, we have not given efficient means for computing them. In particular, it is infeasible both in terms of run time and the size of the resulting citation to go through all rewritings and all assignments within each of them. A precursor for algorithms in this respect is further modeling of (some of) the "black boxes" of the model, which include the citation functions and the semiring operations. We have demonstrated initial ideas in this respect, such as the assumption of idempotence over + and its use in some cases. Even for restricted cases, designing efficient algorithms for computing citations is a non-trivial task. To this end, our future work will also study further connections to relevant classic problems in the literature such as query inclusion (which we have mentioned as useful for the preference relation); query rewriting using views; using logs to understand database usage and decide what citation views should be specified; caching and materialization; and the maintenance and presentation of data provenance.

Acknowledgments The authors would like to thank Peter Buneman and Val Tannen for many fruitful discussions. Peter Buneman initially formulated several of these ideas in the context of XML. This work has been partially funded by NSF IIS 1302212, NSF ACI 1547360, and NIH 3-U01-EB-020954-02S1; by the ERC under the FP7, ERC grant MoDaS, agreement 291071; by the ISF (1636/13); and by a grant from the Blavatnik Interdisciplinary Cyber Research Center.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
2. F. A. Afrati, C. Li, and J. D. Ullman. Using views to generate efficient evaluation plans for queries. *Journal of Computer and System Sciences*, 73(5):703–724, 2007.
3. P. Buneman, S. B. Davidson, and J. Frew. Why data citation is a computational problem. *Communications of the ACM (CACM)*, 59(9):50–57, 2016.
4. 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.
5. S. B. Davidson, D. Deutsch, M. Tova, and G. Silvello. A Model for Fine-Grained Data Citation. In *8th Biennial Conference on Innovative Data Systems Research (CIDR 2017)*, 2017.
6. N. Ferro and G. Silvello. The Road Towards Reproducibility in Science: The Case of Data Citation. In C. Grana and L. Baraldi, editors, *Proc. 13th Italian Research Conference on Digital Libraries (IRCDL 2017)*. Communications in Computer and Information Science (CCIS), Springer, Heidelberg, Germany, 2017.
7. T. J. Green, G. Karvounarakis, and V. Tannen. Provenance Semirings. In L. Libkin, editor, *Proc. of the 26th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 31–40. ACM Press, New York, USA, 2007.