On the Complexity of Query Answering under Access Limitations: A Computational Formalism

Andrea Calí[1,3] and Martín Ugarte[2]

[1]Dept of Comp. Sci. and Inf. Syst.
Birkbeck, Univ. of London, UK

[2]Comp. and Decision Eng. Dept.
Université Libre de Bruxelles

[3]Oxford-Man Inst. of Quantitative Finance
University of Oxford, UK

andrea@dcs.bbk.ac.uk, mugartec@ulb.ac.be

Abstract. The Deep Web is the large corpus of data accessible on the Web through forms and presented in dynamically-generated pages, but not indexable as static pages, and therefore invisible to search engines. Deep Web data are usually modelled as relations with so-called access limitations, that is, they can be queried only by selecting certain attributes. In this paper we give some fundamental complexity results on the problem of processing conjunctive (select-project-join) queries on relational data with access limitations.

1 Introduction

The term *Deep Web* (also called *Hidden Web*) [4, 1] refers to the data content that is created dynamically as the result of a specific search on the web. For example, when we query a web site containing information on films shown at film teathers, the generated output consists of one or more pages containing the result of a query posed on an underlying database; these pages cannot be indexed by search engines. When we query the above hypothetical web site through a form, we are forced to fill in some fields of the form, for instance the City and Title fields; the result is then displayed normally as a table. A Deep Web source can be naturally modelled as a relational table (or a set of relational tables) that can be queried only according to so-called *access patterns*, each of which enforces the selection on some of the attributes (which corresponds to filling the input fields in the form with values), which are called *input* attributes. Relational tables accessible through access patterns are said to have *access limitations*.

Processing structured queries over Deep Web sources is the key problem in the integration of such sources. Interestingly, when Deep Web sources are modelled as relations with access limitations, answering a simple conjunctive (select-project-join) query on such sources requires, in the worst case, the evaluation of a *recursive* Datalog query plan. In such plans, values obtained as output from a source are used as input for other sources; the compatibility of values is established by assigning to each attribute of a relation a so-called *abstract domain*, which expresses the type of value (e.g. name, address etc.) as opposed to the concrete domain (e.g. string, integer etc.).

In this paper we consider the problem of query answering under access limitations in a general way. We believe that the fundamental problem of query answering, even in the case of conjunctive queries (which are, so to say, very vanilla database queries in terms of expressive power), has been largely overlooked. In particular, in the so-called *restricted* case, where the answer has to be computed by querying the sources according to the limitations and *without any prior knowledge of the underlying data*, it is not yet clear how to formally model the fact that part of the input is *partially hidden* to the algorithm that is to compute the answers, as the data can be accessed only according to the access limitations. Therefore we offer the following contributions.

- We provide a formalisation of the query answering problem where the problem has a *hidden input* (the data) and an *open input* (the query and the schema with access limitations).
- We propose a computational model for the query answering problem consisting of a Turing machine which queries an *oracle machine* (a transducer in this case) that, given the information for an access to the hidden relational data, writes the corresponding output.
- We revisit some fundamental results, and propose some new ones, on conjunctive query answering in the new framework, so as to set the basis for future investigation, in particular regarding more expressive query languages.

2 Modelling Query Answering

We assume the reader is familiar with the notions of relational schema and instance, conjunctive query and Datalog program. We consider relational schemata with access limitations, where the limitations consist of annotations on predicates (or columns, or attributes) that express whether each argument/attribute/column is *input* (needs to be selected) or *output*; for instance, r^{iio}, of arity 3, has the first two attributes as input attributes, and the third as output — in order to query r, two input values are to be provided as selection values on the first two attributes. Suppose the schema of r is $r(A, A, B)$, where each attribute name corresponds to a so-called *abstract domain*, associated with a real-world domain (such as car registration number, person name, phone number etc.) that is more abstract than the underlying concrete domain (such as string, integer, date etc.). In order to access r (or more precisely its instance within a database instance D) one needs to provide a pair $\langle a_1, a_2 \rangle$ of constants of the abstract domain D_A of A, that is $\{a_1, a_2\} \subseteq D_A$. The result of the access/query consists of all facts $r(a_1, a_2, b)$ contained in $r(D)$, where $r(D)$ is the instance of r in the database D.

In the presence of access limitations on the sources, queries cannot be usually evaluated as in the traditional case. Given a conjunctive query q, a schema with access limitations (implicit), a database D and a set I of initial constants, the answers to q, denoted $\text{ans}(q, I, D)$, are obtained starting from the constants in I and extracting all possible tuples (by using the constants as input in all possible ways); with the newly obtained constants again all possible tuples are extracted, and so on, until no new tuple is extracted – see e.g. [1]. The above procedure is naturally encoded in a Datalog program [4, 1] that encodes the fact that suitable

inputs are to be fed to the sources with limitations. However, with the Datalog encoding at hand, in principle we are not sure whether a better strategy is possible. We argue that we cannot consider the query answering problem from the computational point of view without going beyond the aforementioned Datalog encoding. We address the above issues by providing a formal framework for the problem of query answering under access limitations.

We start by considering a Turing machine as our obvious computational tool. In order to model the relations with access limitations we use a special type of *oracle Turing machine (OTM)* as follows: *(1)* the OTM, apart from the standard tape, has an *oracle tape*; *(2)* to access a source, the OTM writes the input (the source relation name R plus n constants, where n is the number of input attributes or R), enters in a special state q_A (*ask* state); then the oracle writes the output on the oracle tape and the OTM goes into state q_R (*response* state). Notice that the oracle in this case does not solve a problem of a certain complexity class in order to improve the computational power of the main OTM; instead, the oracle serves *to hide part of the input* (the database D) from the main computation that is carried out by the main OTM. We need in this case to split the input of the problem into two parts: an *open* input and a *secret* input. The open input is the standard input to the OTM, which goes on the tape; the hidden input instead is accessible to the oracle but not to the OTM.

Considering the main OTM $\mathcal{M}$ as an acceptor, we can define complexity classes of the type $\mathcal{C}^{[\mathrm{RL}]}$ (where RL stands for "Relational with Limitations"), where $\mathcal{C}^{[\mathrm{RL}]}$ is the class of languages accepted with complexity $\mathcal{C}$ by an OTM having an oracle that provides access to relational sources with access limitations as above described.

Notice that the problem solved by the oracle is, in the absence of additional data structures, linear in the size of the database D. However, replacing the oracle with an arbitrary linear time transducer would destroy the purpose of the oracle as it is. In such a case, even in the presence of secret input (as we are in this context), the oracle could simply return the whole D at the first access, thus making the hidden input *de facto* open instead.

The availability of a hidden input and an OTM opens novel scenarios. For instance, we can formalise the problem of guessing a secret number between 0 and 2^{n-1}. Consider a schema constituted by a single n-ary relational predicate r with all input attributes, and a database D as *secret* input, constituted of a single tuple $r(c_0, \ldots, c_{n-1})$ where the c_i are either 0 or 1. Now consider the set $I = \{0, 1\}$ of initial constants as well as the Boolean CQ q defined as $q() \leftarrow r(X_0, \ldots, X_{n-1})$. Obviously D represents a number between 0 and 2^{n-1} in binary notation, and the query q is true, but the only way the OTM is able to compute the answer to q is by accessing r with the right number (in binary, as a sequence of values in I, that is 0 or 1). It turns out that guessing the number can be done in polynomial time with a non-deterministic OTM, but that instead a deterministic OTM will take exponential time in the worst case (assuming the schema is not fixed). This suggests that, under a RL oracle and hidden input, there is a problem that can be decided in non-deterministic polynomial time but not in deterministic polynomial time, that is $\mathrm{P}^{[\mathrm{RL}]} \neq \mathrm{NP}^{[\mathrm{RL}]}$. We argue that this observation shows clearly that there is a fundamental difference between studying access limitations under traditional Turing machines and OTMs, due

case no.	query	database	schema	det. upper	non-det. upper
#1	atomic	variable	variable	$\mathrm{EXP}^{[\mathrm{RL}]}$	$\mathrm{NP}^{[\mathrm{RL}]}$
#2	CQ	variable	variable	$\mathrm{EXP}^{[\mathrm{RL}]}$	$\mathrm{NP}^{[\mathrm{RL}]}$
#3	atomic	variable	fixed	$\mathrm{P}^{[\mathrm{RL}]}$	♠
#4	atomic	fixed	variable	$\mathrm{linear}^{[\mathrm{RL}]}$	♠

Fig. 1. Query answering: upper bounds. Uninteresting cases are marked with ♠.

to the necessity of dealing with the hidden input. In fact, when there is no hidden input the problem of checking whether a relation is non-empty is trivial; on the contrary, if there is a hidden input, the example above shows that one needs at least deterministic exponential time.

3 Query Answering

In this section we revisit some results [3, 2], and present novel ones, on Boolean CQ answering under access limitations, using our new formalism. Due to space limitations we summarise our results in Figure 1. We assume the database D to be the only secret input in all cases. We show selected upper bounds for deterministic and non-deterministic OTMs.

Notice that (cases #1 and #2) we cannot do better than non-deterministic polynomial time or deterministic exponential time when both the database D and the schema (in particular, the maximum arity W of predicates) are variable inputs; this is a consequence of our observations at the end of Section 2 Such complexity absorbs the inherent intractability of CQ evaluation. Differently, in cases #3 and #4, the fixed D or W cause the number of accesses to the oracle to be less than exponential, hence we get lower complexity bounds.

Given that the relationship between complexity classes changes in the presence of an oracle lower bounds are not at all obvious. Beyond this taster paper, we plan to further investigate fundamental problems of query answering in this setting, considering complexity classes of the kind $\mathcal{C}^{[\mathrm{RL}]}$ based on our OTM which, we argue, is the most suitable tool to investigate this topic.

Acknowledgments. Andrea Calí acknowledges partial support by the EPSRC project "Logic-based Integration and Querying of Unindexed Data" (EP/E010865/1)" as well as from the Birkbeck BEI School Grant "Exposing the Deep Web in the Linked Data Cloud".

References

1. Andrea Calì and Davide Martinenghi. Querying data under access limitations. In *Proc. of ICDE*, pages 50–59, 2008.
2. Andrea Calí, Davide Martinenghi, Igor Razgon, and Martín Ugarte. Querying the deep web: Back to the foundations. In *Proc. of AMW*, 2017.
3. Andrea Calí and Igor Razgon. Complexity of conjunctive query answering under access limitations (preliminary report). In *Proc. of SEBD*, pages 256–263, 2014.
4. Chen Li and Edward Chang. Answering queries with useful bindings. *ACM Transactions on Database Systems*, 26(3):313–343, 2001.