Some Research Directions in Consistent Query Answering: A Vision

Leopoldo Bertossi[⋆]
Carleton University, School of Computer Science
Ottawa, Canada
bertossi@scs.carleton.ca

Some Past Research

Research in *consistent query answering* (CQA) in databases was initiated in the database community with the publication of [1], where the main goal was to formalize the notion of *consistent answer* to a query posed to a possibly *inconsistent* database, i.e. that fails to satisfy a given set of integrity constraints (ICs) that are not enforced by the system.

For many reasons [9], such inconsistencies may naturally arise. Consistent answers were semantically characterized as those answers that can be obtained, as normal answers, from all the possible minimally repaired versions of the inconsistent database at hand. According to [1], a *repair* of a relational database instance D is an instance that satisfies the ICs, with the same schema as D, that in set theoretic terms, minimally differ from D wrt whole tuples that are either deleted or inserted in order to restore consistency. Then, the consistent answers to a query are those that are invariant under repairs.

Since computing consistent answers by appealing directly to the definition, i.e. via explicit computation and querying of all possible repairs, is [9] practically unfeasible, [1] introduced the first mechanism for computing consistent answers to first-order queries that did not appeal to explicit computation of repairs. Instead, the idea was to modify the original query without changing the inconsistent database, then pose the rewritten query, and collect the normal answers to it. This mechanism turned out to be sound and/or complete for several classes of queries and ICs that were identified in [1]. This form of query rewriting works for some useful classes of queries and ICs, but its applicability is still limited.

With the purpose of computing consistent answers to full first-order queries, in [5, 19] the repairs of a database were characterized as the stable models of disjunctive logic programs. In consequence, obtaining consistent answers became the problem of computing answers from the repair program combined with the query program under the skeptical stable model semantics. Since the query program is quite general, the queries supported could be found much beyond first-order, actually in rich extensions of Datalog. However, the number of atoms in the ICs may produce a blow-up in the number of program rules.

The semantics of consistent query answers to scalar *aggregate queries* wrt functional dependencies was introduced and studied in [3]. Since those queries

[⋆] Also Faculty Fellow of the IBM Center for Advanced Studies (Toronto Lab.)

return numerical values, the natural semantics is the *range semantics*, i.e. based on the shortest numerical interval that contains the answers to the query from all possible repairs. Algorithms were given to compute the range semantics for the (provably) tractable cases, and the untractable cases were fully characterized [4].

The semantics of CQA was further studied in a non-classical logical system, the *annotated predicate logic* (APC). As a result, database repairs were characterized as a special class of minimal models of a theory written in a particular version of APC (the truth-lattice is a parameter in APC, so the right lattice for this task was identified). This turned the problem of obtaining consistent answers to an arbitrary first-order query into a problem of non-monotonic reasoning from an APC theory [apc,nmr], for which there are no implementations available.

The results obtained using answer set programming and the theoretical framework obtained via APC motivated trying to mimic APC reasoning using answer set programming. So, the truth annotations introduced in [2] were used as distinguished domain constants to be used in database atoms extended with an extra argument. Reasoning with annotations and implicit interaction between them in the APC version was made explicit in answer set programming [7, 8, 12].

In this way, it is possible to capture every IC, no matter how many atoms in it, as a single rule in the program, avoiding the exponential blow-up [5]. The stable models of the generated disjunctive logic program (with program denial constraints) were established to be in one-to-one correspondence with the repairs [8]. So, CQA could be done for any kind of universal ICs (i.e. no existential quantifiers in it) and any query expressed in extensions of Datalog.

In [12] referential ICs (they contain existential quantifiers) are considered, and the programs introduced in [7] were extended accordingly, assuming that referential ICs could be repaired through tuple deletions or insertions of null values that are not propagated through other ICs. The other edge of the problem, also addressed in [12], but never considered before, is that the original database could already be *incomplete*, i.e. containing null values. In consequence, their presence has to be considered when defining repairs.

In [8, 14] it is shown how the logic programming approach to CQA can be made more efficient by applying several optimizations, like pruning unnecessary program rules, rule transformation to capture cases of lower complexity (e.g. head-cycle-free), optimizing query evaluation using *magic sets* techniques, optimizing the access to the underlying database, etc.

In between, a much more clear picture of the complexity of CQA has emerged. Also tractable classes have been identified and implementations developed [15, 13, 18]. To the best of our knowledge, all the algorithms and implementations available compute consistent answers to queries from scratch, except for the possible precomputation of all the repairs (or stable models in the case of logic programs).

Looking Forward

Many problems are still open in the area of consistent query answering. Many of them, interesting, challenging, but also specific to particular techniques, applications, variations of the basic notions, and implementation. However, a general and still open problem is to achieve a global understanding of the "logic of consistent query answering", i.e. the logic that governs the definition and computation of consistent query answers, and reasoning with and about them.

We know, e.g. that CQA follows a non-monotonic logic [9]. We also know that it is, in some sense, a form of modal logic (being the repairs the possible worlds). We can see that it is not compositional as classical query answering in databases, e.g. the answer set to a conjunctive query may not be the intersection of the answer sets, etc. We do not have a complete knowledge of its logic or their properties. In particular, *compositionality* of CQA has not been investigated. We do not know what is correct or what can be used for query answering in that direction yet. This is an objective that deserves much research.

Let us recall that classical query answering in relational databases follows, essentially, a first-order logic, whose most notorious expression is the relational calculus. From this point of view, it is semantically clear what is an answer to a query and how answers to queries can be combined in order to give answers to more complex queries. This is because, the notion of truth in first-order logic has nice compositional properties, as established by its Tarskian semantics. Already the above mentioned non-monotonicity makes CQA depart from first-order logic, that is monotonic.

Non-monotonic formalisms have been used to characterize and compute CQA, e.g. annotated predicate logic [2], logic programs with stable model semantics [5, 7], non-monotonic analytic tableaux [10], circumscription [10]. However not much emphasis has been placed on the study of the *intrinsic logic* of consistent query answering. There are natural open questions in this direction: (a) How can we classify the underlying non-monotonic logic? (b) What kind of modal logic we have? With what kind of accessibility relation? Can it be axiomatized? (c) What compositionality properties it has? (d) Is there a set-theoretic, algebraic counterpart (like relational algebra to relational calculus)? (d) What are the connections to other logics that have been used to capture and formalize ICs in databases. Addressing all these issues becomes more interesting and difficult if one considers that the database may be incomplete, and the database community, including database practice, is far from having an agreement on the semantics of incomplete databases.

The repair semantics that has been intensively studied is the one introduced in [1]. However, other repair semantics have been considered: Repairs that minimally differ in cardinality from the original database [9, 5], and repairs that minimize some aggregation function over the differences of attribute values between the repair and the original instance [17, 20, 11, 16]. It should be clear that any choice of a repair semantics will have an effect on the underlying logic of CQA. We think that identifying general properties of the *reasonable* repair semantics and studying their impact on the logic of CQA is a very important re-

search direction. Unifying principles seem to be necessary at this stage, in order to have a better understanding of CQA, both in theoretical and practical terms.

References

[1] M. Arenas, L. Bertossi and J. Chomicki. Consistent Query Answers in Inconsistent Databases. *Proc. ACM Symposium on Principles of Database Systems (PODS 99)*, ACM Press, 1999, pp. 68–79.

[2] M. Arenas, L. Bertossi and M. Kifer. Applications of Annotated Predicate Calculus to Querying Inconsistent Databases. In *Computational Logic-CL2000*, J. Lloyd et al. (eds.). Springer LNAI 1861, 2000, pp. 926–941.

[3] M. Arenas, L. Bertossi and J. Chomicki. Scalar Aggregation in FD-Inconsistent Databases. In *Database Theory - ICDT 2001*, Springer LNCS 1973, 2001, pp. 39-53.

[4] M. Arenas, L. Bertossi, J. Chomicki, X. He, J. Spinrad and V. Raghavan. Scalar Aggregation in Inconsistent Databases. *Theoretical Computer Science*, 2003, 296(3):405-434.

[5] M. Arenas, L. Bertossi and J. Chomicki. Answer Sets for Consistent Query Answers. *Theory and Practice of Logic Programming*, 2003, 3(4&5):393-424.

[6] P. Barcelo and L. Bertossi. Repairing Databases with Annotated Predicate Logic. *Proc. Nineth International Workshop on Non-Monotonic Reasoning (NMR 02)*, S. Benferhat and E. Giunchiglia (eds.), 2002, pp. 160–170.

[7] P. Barcelo and L. Bertossi. Logic Programs for Querying Inconsistent Databases. In *Practical Applications of Declarative Languages (PADL 03)*, Springer LNCS 2562, 2003, pp. 208-222.

[8] P. Barcelo, L. Bertossi and L. Bravo. Characterizing and Computing Semantically Correct Answers from Databases with Annotated Logic and Answer Sets. In *Semantics in Databases*, Springer LNCS 2582, 2003, pp. 1–27.

[9] L. Bertossi and J. Chomicki. Query Answering in Inconsistent Databases. In *Logics for Emerging Applications of Databases*, J. Chomicki, G. Saake and R. van der Meyden (eds.), Springer, 2003, pp. 43-83.

[10] L. Bertossi and C. Schwind. Analytic Tableaux for Querying Inconsistent Databases. *Annals of Mathematics and Artificial Intelligence*, 2004, 40(1-2):5-35.

[11] L. Bertossi, L. Bravo, E. Franconi and A. Lopatenko. Complexity and Approximation of Fixing Numerical Attributes in Databases Under Integrity Constraints. *Proc. of the Databases Programming Languages Conference (DBPL 05)*, Springer LNCS 3774, 2005, pp. 262-278.

[12] L. Bravo and L. Bertossi. Consistent Query Answering under Inclusion Dependencies. *Proc. 14th Annual IBM Centers for Advanced Studies Conference (CASCON 04)*, 2004, pp. 202-216.

[13] Cali, A., Lembo, D. and Rosati, R. On the Decidability and Complexity of Query Answering over Inconsistent and Incomplete Databases. *Proc. ACM Symposium on Principles of Database Systems (PODS 03)*, ACM Press, 2003, pp. 260-271.

[14] M. Caniupan and L. Bertossi. Optimizing Repair Programs for Consistent Query Answering. *Proc. International Conference of the Chilean Computer Science Society (SCCC 05)*, IEEE Computer Society Press, 2005, pp. 3-12.

[15] J. Chomicki and J. Marcinkowski. Minimal-Change Integrity Maintenance Using Tuple Deletions. *Information and Computation*, 2005, 197(1-2):90-121.

[16] S. Flesca, F. Furfaro and F. Parisi Consistent Query Answers on Numerical Databases under Aggregate Constraints. *Proc. Tenth International Symposium on Database Programming Languages (DBPL 05)*, Springer LNCS 3774, 2005, pp. 279-294.

[17] E. Franconi, A. Laureti Palma, N. Leone, S. Perri and F. Scarcello. Census Data Repair: a Challenging Application of Disjunctive Logic Programming. In *Proc. Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 01)*, Springer LNCS 2250, 2001, pp. 561-578.

[18] A. Fuxman and R. Miller. First-Order Query Rewriting for Inconsistent Databases. In *Proc. ICDT*, Springer LNCS 3363, 2005, pp. 337-354.

[19] G. Greco, S. Greco and E. Zumpano. A Logical Framework for Querying and Repairing Inconsistent Databases. *IEEE Transactions in Knowledge and Data Engineering*, 2003, 15(6):1389-1408.

[20] J. Wijsen Condensed Representation of Database Repairs for Consistent Query Answering. *Proc. International Conference on Database Theory (ICDT 03)*, Springer LNCS 2572, 2003, pp. 378-393.