The Shapley Value of Tuples in Query Answering

Ester Livshits Leopoldo Bertossi Benny Kimelfeld Moshe Sebag

Various measures have been proposed for *quantifying the contribution of a database fact (tuple) f to a query result*. Meliou et al. [8] adopted the quantity of *responsibility* that is inversely proportional to the minimal number of facts that should be removed to make f counterfactual (i.e., removing f transitions the query answer from true to false). This measure, however, is fundamentally designed for non-numerical queries, and it is not clear whether it can incorporate the numerical contribution of f. Salimi et al. [13] proposed the *causal effect*: assuming the facts are randomly removed independently and uniformly, what is the difference in the expected query result between assuming the presence and the absence of f? In this work, we investigate the application of a well known measure from cooperative game theory to quantifying the contribution of facts to query results. We believe that the suitability of the measure for this task is backed by its theoretical justification as well as its massive adoption in a plethora of fields.

The Shapley value was introduced by Lloyd Shapley in a seminal 1952 article [14]. He considered a *cooperative game* that is played by a set A of players and is defined by a *wealth function* v that assigns, to each coalition $S \subseteq A$, the wealth $v(S)$. For instance, A might be a set of politicians, and $v(S)$ the number of votes that a poll assigns to the party that consists of the candidates in S. The question is how to distribute the wealth $v(A)$ among the players, or from a different perspective, how to quantify the contribution of each player to the overall wealth. Shapley considered distribution functions that satisfy a few axioms of a good behavior. Quite remarkably, he has established that there is a *single* function that satisfies all of them, and this function has become known as the *Shapley value*.[1] The Shapley value has been applied in various areas and fields beyond cooperative game theory [5–7, 9–11].

In this work, we apply the Shapley value to quantifying the contribution of facts to query results. As in previous work on quantification of contribution of facts [8, 13], we view the database D as consisting of two types of facts. *Exogenous* facts are taken as given without questioning, and are beyond experimentation with counterfactual scenarios. *Endogenous* facts are those we may have control over, and we reason about existence and marginal contribution for these facts. Our focus here is on numerical queries (i.e., queries that map databases to real numbers), and specifically on *Boolean Conjunctive Queries* (BCQs)[2] and *aggregates* over CQs. A CQ is an expression of the form $q(\vec{x})$:- $R_1(\vec{t_1}), \ldots, R_n(\vec{t_n})$, where each R_i is a relation symbol, each $\vec{t_i}$ is a tuple of variables and constants (the arity of $\vec{t_i}$ should match the number of attributes in R_i), and $\vec{x}$ is a tuple of variables from $\vec{t_1}, \ldots, \vec{t_n}$. The answers to q on a database D are the tuples $\vec{c}$ that are obtained by projecting to $\vec{x}$ all homomorphisms from q to D. If $\vec{x}$ has arity zero, then the query is Boolean, in which case we denote by $D \models q$ the fact that D satisfies q. Here, we view a BCQ as the numerical query q defined by $q(D) = 1$ if $D \models q$ and $q(D) = 0$ if $D \not\models q$. An aggregate query is a CQ q followed by an aggregate function that maps $q(D)$ to a real number (e.g., count, sum, max, average). The core computational problem for a query is then: given a database and an endogenous fact, compute the Shapley value of the fact.

The Shapley value of a fact f in a database D w.r.t. a numerical query α is formally defined as follows:

$$\mathrm{Shapley}(D, \alpha, f) = \sum_{E \subseteq D_{\mathsf{n}} \setminus \{f\}} \frac{|E|! \cdot (|D_{\mathsf{n}}| - |E| - 1)!}{|D_{\mathsf{n}}|!} \Big(\alpha(D_{\mathsf{x}} \cup E \cup \{f\}) - \alpha(D_{\mathsf{x}} \cup E) \Big) \tag{1}$$

where D_{n} and D_{x} are the sets of endogenous and exogenous facts in D, respectively. Note that $|E|! \cdot (|D_{\mathsf{n}}| - |E| - 1)!$ is the number of permutations over D_{n} such that all facts in E come first, then f, and then all remaining facts. The Shapley value is essentially a probabilistic expectation. Suppose that we form a database by taking the facts one by one, randomly and uniformly without replacement; while doing so, we record the change of α due to the addition of f as the random contribution of f. Then the Shapley value of f is the expectation of the random contribution.

Our main result is a full classification of (i.e., a dichotomy in) the data complexity of the problem of computing $\mathrm{Shapley}(D, q, f)$ for BCQs without self-joins, where a self-join is a pair of distinct atoms over the same relation

[1] For further reading, we refer the reader to the book by Roth [12].

[2] Focusing on Boolean queries allows us to keep the presentation considerably simpler while, at the same time, retaining the fundamental challenges. Our results can be easily extended to to general CQs [1], by considering each query answer separately.

symbol. As we show, the classification criterion is the same as that of query evaluation over tuple-independent probabilistic databases [3]; however, it is not clear whether and/or how we can use their dichotomy to prove ours, in each of the two directions (tractability and hardness). We first introduce the definition of a *hierarchical* query. We say that a BCQ q is hierarchical if for all variables x and x' in q it holds that $A_x \subseteq A_{x'}$, or $A_{x'} \subseteq A_x$, or $A_x \cap A_{x'} = \emptyset$, where A_y is the set of atoms $R_i(\vec{t_i})$ of q that contain y (that is, y occurs in $\vec{t_i}$) [2]. We have the following.

Theorem 1 *Let q be a Boolean CQ without self joins. If q is hierarchical, then* $\mathrm{Shapley}(D, q, f)$ *can be computed in polynomial time, given D and f. Otherwise, the problem is* $\mathrm{FP}^{\#\mathrm{P}}$*-complete.*

Recall that $\mathrm{FP}^{\#\mathrm{P}}$ is the class of functions computable in polynomial time with an oracle to a $\#\mathrm{P}$-complete problem (e.g., counting the number of satisfying assignments of a propositional formula). This complexity class is considered intractable, and is known to be above the polynomial hierarchy (Toda's theorem [15]).

For the positive side of Theorem 1, observe that the computation of $\mathrm{Shapley}(D, q, f)$ easily reduces to the problem of counting the sets of size k of endogenous facts that, along with the exogenous facts, satisfy q. More formally, the reduction is to the problem of computing $|\mathrm{Sat}(D, q, k)|$ where $\mathrm{Sat}(D, q, k)$ is the set of all subsets E of D_n such that $|E| = k$ and $D_\mathsf{x} \cup E \models q$. The reduction is as follows.

$$
\begin{aligned}
\mathrm{Shapley}(D, q, f) &= \sum_{E \subseteq (D_\mathsf{n} \setminus \{f\})} \frac{|E|!(|D_\mathsf{n}| - |E| - 1)!}{|D_\mathsf{n}|!} \Big(q(D_\mathsf{x} \cup E \cup \{f\}) - q(D_\mathsf{x} \cup E) \Big) \\
&= \sum_{E \subseteq (D_\mathsf{n} \setminus \{f\})} \frac{|E|!(|D_\mathsf{n}| - |E| - 1)!}{|D_\mathsf{n}|!} \Big(q(D_\mathsf{x} \cup E \cup \{f\}) \Big) - \sum_{E \subseteq (D_\mathsf{n} \setminus \{f\})} \frac{|E|!(|D_\mathsf{n}| - |E| - 1)!}{|D_\mathsf{n}|!} \Big(q(D_\mathsf{x} \cup E) \Big) \\
&= \left(\sum_{k=0}^{|D_\mathsf{n}|-1} \frac{k!(|D_\mathsf{n}| - k - 1)!}{|D_\mathsf{n}|!} \times |\mathrm{Sat}(D', q, k)| \right) - \left(\sum_{k=0}^{|D_\mathsf{n}|-1} \frac{k!(|D_\mathsf{n}| - k - 1)!}{|D_\mathsf{n}|!} \times |\mathrm{Sat}(D \setminus \{f\}, q, k)| \right)
\end{aligned}
$$

In the last expression, D' is the same as D, except that f is viewed as *exogenous* instead of *endogenous*. Hence, whenever $|\mathrm{Sat}(D, q, k)|$ can be computed in polynomial time, the Shapley value can also be computed in polynomial time. And, indeed, $|\mathrm{Sat}(D, q, k)|$ is computable in polynomial time for hierarchical queries q. As expected for a hierarchical query, our algorithm is a recursive procedure that acts differently in three different cases: *(a)* q has no variables (only constants), *(b)* there is a variable x (called a *root variable*) that occurs in all atoms of q, or *(c)* q consists of two (or more) sub-queries that do not share any variables. Since q is hierarchical, at least one of these cases always apply [4]. The algorithm is fairly straightforward, except for case (b) where there is a root variable, and then we combine the recursive call with dynamic programming.

For the negative side of the theorem, membership in $\mathrm{FP}^{\#\mathrm{P}}$ is straightforward, so we omit the discussion on that. Similarly to Dalvi and Suciu [3], our proof of hardness consists of two steps. First, we prove the $\mathrm{FP}^{\#\mathrm{P}}$-hardness of computing $\mathrm{Shapley}(D, \mathsf{q}_{\mathsf{RST}}, f)$, where $\mathsf{q}_{\mathsf{RST}}() \text{ :- } R(x), S(x, y), T(y)$. Second, we reduce the computation of $\mathrm{Shapley}(D, \mathsf{q}_{\mathsf{RST}}, f)$ to the problem of computing $\mathrm{Shapley}(D, q, f)$ for any non-hierarchical BCQ q without self-joins. We focus on the first step, which is the most intricate part of the proof. To prove that computing $\mathrm{Shapley}(D, \mathsf{q}_{\mathsf{RST}}, f)$ is $\mathrm{FP}^{\#\mathrm{P}}$-complete, we construct a (Turing) reduction from the problem of computing the number $|\mathsf{IS}(g)|$ of independent sets of a given bipartite graph g. Given an input bipartite graph $g = (V, E)$ for which we wish to compute $|\mathsf{IS}(g)|$, we construct $n + 1$ different input instances (D_j, f), for $j = 1, \ldots, n + 1$, of the problem of computing $\mathrm{Shapley}(D_j, \mathsf{q}_{\mathsf{RST}}, f)$, where $n = |V|$. Each instance provides us with an equation over the numbers $|\mathsf{IS}(g, k)|$ of independent sets of size k in g for $k = 0, \ldots, n$. We then show that the set of equations constitutes a non-singular matrix that, in turn, allows us to extract the $|\mathsf{IS}(g, k)|$ in polynomial time (e.g., via Gaussian elimination). This is enough, since $|\mathsf{IS}(g)| = \sum_{k=0}^{n} |\mathsf{IS}(g, k)|$.

As mentioned above, the Shapley value is a probabilistic expectation. The linearity of expectation allows us to extend our dichotomy to arbitrary summations (including counting) over CQs without self-joins. In fact, the hardness side of Theorem 1 generalizes to all aggregates over CQs. The only exception is when α is a *constant* numerical query. The general conclusion is that computing the exact Shapley value is notoriously hard, but the picture is far more optimistic if approximation is allowed under strong guarantees of error boundedness. For non-hierarchical queries (and, in fact, all unions of CQs), the Shapley value is approximable (i.e., has a multiplicative *Fully Polynomial-Time Approximation Scheme*) via Monte Carlo sampling. This approximation result also generalizes to an FPRAS for summations over CQs.

References

[1] S. Cohen, W. Nutt, and Y. Sagiv. Deciding equivalences among conjunctive aggregate queries. *J. ACM*, 54(2):5, 2007.

[2] N. N. Dalvi, C. Ré, and D. Suciu. Probabilistic databases: diamonds in the dirt. *Commun. ACM*, 52(7):86–94, 2009.

[3] N. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, pages 864–875. Morgan Kaufmann, 2004.

[4] N. N. Dalvi and D. Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6):30:1–30:87, 2012.

[5] F. Gul. Bargaining foundations of shapley value. *Econometrica: Journal of the Econometric Society*, pages 81–95, 1989.

[6] Z. Liao, X. Zhu, and J. Shi. Case study on initial allocation of shanghai carbon emission trading based on shapley value. *Journal of Cleaner Production*, 103:338–344, 2015.

[7] R. T. Ma, D. M. Chiu, J. Lui, V. Misra, and D. Rubenstein. Internet economics: The use of shapley value for isp settlement. *IEEE/ACM Transactions on Networking (TON)*, 18(3):775–787, 2010.

[8] A. Meliou, W. Gatterbauer, K. F. Moore, and D. Suciu. The complexity of causality and responsibility for query answers and non-answers. *PVLDB*, 4(1):34–45, 2010.

[9] R. Narayanam and Y. Narahari. A shapley value-based approach to discover influential nodes in social networks. *IEEE Transactions on Automation Science and Engineering*, 8(1):130–147, 2011.

[10] T. Nenova. The value of corporate voting rights and control: A cross-country analysis. *Journal of financial economics*, 68(3):325–351, 2003.

[11] L. Petrosjan and G. Zaccour. Time-consistent shapley value allocation of pollution cost reduction. *Journal of economic dynamics and control*, 27(3):381–398, 2003.

[12] A. E. Roth. *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[13] B. Salimi, L. E. Bertossi, D. Suciu, and G. V. den Broeck. Quantifying causal effects on query answering in databases. In *TAPP*, 2016.

[14] L. S. Shapley. A value for n-person games. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, Princeton, 1953.

[15] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.