Understanding the Complexity of Lifted Inference and Asymmetric Weighted Model Counting

Extended Abstract of [3]

Eric Gribkoff
University of Washington
eagribko@cs.uw.edu

Guy Van den Broeck
UCLA, KU Leuven
guyvdb@cs.ucla.edu

Dan Suciu
University of Washington
suciu@cs.uw.edu

ABSTRACT

We highlight our work on lifted inference for the asymmetric Weighted First-Order Model Counting problem (WFOMC), which counts the assignments that satisfy a given sentence in first-order logic. This work is at the intersection of probabilistic databases and statistical relational learning. First, we discuss how adding negation can lower the query complexity, and describe the essential element (resolution) necessary to extend a previous algorithm for positive queries to handle queries with negation. Second, we highlight the difficulties with extending the dichotomy result for positive queries to include negation. We outline how we overcame these difficulties to establish a novel dichotomy for a non-trivial fragment of first-order CNF sentences with negation. Finally, we discuss directions for future work.

1. OVERVIEW

Our work is concerned with weighted first-order model counting (WFOMC), where we sum the weights of assignments that satisfy a sentence (or query) in finite-domain first-order logic. This reasoning task underlies efficient algorithms for probabilistic reasoning in AI, with statistical relational representations such as Markov logic [9, 2] and probabilistic programs [8]. Moreover, WFOMC uncovers a deep connection between AI and database research, where query evaluation in *probabilistic databases* (PDBs) [6] essentially considers the same task. A PDB defines a probability, or weight, for every possible world, and each database query is a sentence encoding a set of worlds, whose combined probability we want to compute. We refer to our setting as *asymmetric* WFOMC, because it allows the weights of each atom to be distinct. This subsumes the symmetric setting considered in AI, where probabilities are set per relation.

We introduce a new algorithm $\mathbf{Lift}^{\mathbf{R}}$ that extends the algorithm of [1]. This latter algorithm only supports queries without negation, but comes with strong theoretical guarantees in the form of a dichotomy theorem. $\mathbf{Lift}^{\mathbf{R}}$ applies to general CNF queries, with arbitrary negation, by introducing a resolution operation to discover new prime implicates. As we show below, this step is crucial to proving completeness of the algorithm. Our algorithm performs *lifted inference*, meaning that it exploits the relational structure of the query. Moreover, it performs domain-lifted inference, because it runs in time polynomial in the database size [7].

2. NEGATION CAN LOWER COMPLEXITY

The presence of negations can lower a query's complexity, and $\mathbf{Lift}^{\mathbf{R}}$ exploits this. To see this, consider the query

$$\begin{aligned} Q &= \forall x \forall y\ (\mathtt{Tweets}(x) \vee \neg \mathtt{Follows}(x,y)) \\ &\quad \wedge \forall x \forall y\ (\mathtt{Follows}(x,y) \vee \neg \mathtt{Leader}(y)). \end{aligned}$$

The query says that if x follows anyone then x tweets, and that everybody follows the leader[1].

Our goal is to compute the probability $\Pr(Q)$, knowing the probabilities of all (ground) atoms in the domain. We note that the two clauses are dependent (since both refer to the relation `Follows`), hence we cannot simply multiply their probabilities; in fact, we will see that if we remove all negations, then the resulting query is #P-hard; the algorithm described by [1] would immediately get stuck on this query. Instead, $\mathbf{Lift}^{\mathbf{R}}$ takes advantage of the negation, by first computing the prime implicate

$$\forall x\ \mathtt{Tweets}(x) \vee \forall y\ \neg \mathtt{Leader}(y),$$

which is a disconnected clause (the two literals use disjoint logical variables, x and y respectively). After applying distributivity we obtain:

$$\begin{aligned} Q &= (Q \wedge (\forall x\ \mathtt{Tweets}(x))) \vee (Q \wedge (\forall y\ \neg \mathtt{Leader}(y))) \\ &= Q_1 \vee Q_2 \end{aligned}$$

and $\mathbf{Lift}^{\mathbf{R}}$ applies the inclusion-exclusion formula:

$$\Pr(Q) = \Pr(Q_1) + \Pr(Q_2) - \Pr(Q_1 \wedge Q_2)$$

After simplifying the three queries, they become:

$$\begin{aligned} Q_1 &= \forall x \forall y\ (\mathtt{Follows}(x,y) \vee \neg \mathtt{Leader}(y)) \wedge \forall x\ (\mathtt{Tweets}(x)) \\ Q_2 &= \forall x \forall y\ (\mathtt{Tweets}(x) \vee \neg \mathtt{Follows}(x,y)) \wedge \forall y\ (\neg \mathtt{Leader}(y)) \\ Q_1 \wedge Q_2 &= \forall x\ (\mathtt{Tweets}(x)) \wedge \forall y\ (\neg \mathtt{Leader}(y)) \end{aligned}$$

The probability of Q_1 can now be obtained by multiplying the probabilities of its two clauses; same for the other two queries. As a consequence, our algorithm computes the

[1] To see this, rewrite the query as $(\mathtt{Follows}(x,y) \Rightarrow \mathtt{Tweets}(x)) \wedge (\mathtt{Leader}(y) \Rightarrow \mathtt{Follows}(x,y))$.

probability $\Pr(Q)$ in polynomial time in the size of the domain and the probabilistic database.

If we remove all negations from Q and rename the predicates we get the following query:

$$h_1 \;=\; \forall x \forall y\ (R(x) \vee S(x,y)) \wedge (S(x,y) \vee T(y))$$

It was proven in [1] that computing the probability of the dual of h_1 is #P-hard in the size of the PDB. Thus, the query Q with negation is *easy*, while h_1 is hard, and $\mathbf{Lift^R}$ takes advantage of this by applying resolution to find the disconnected prime implicate $\forall x \forall y\ \mathtt{Tweets}(x) \vee \neg \mathtt{Leader}(y)$.

3. A DICHOTOMY FOR TYPE-1 QUERIES

We prove a novel dichotomy for a subclass of CNF queries. First we review an earlier result, to put ours in perspective. In [1], an algorithm and dichotomy result for positive queries is shown. This result can be adapted to show that $\mathbf{Lift^R}$ restricted to monotone (i.e., negation-free) queries admits a dichotomy, and the following theorem continues to hold.

THEOREM 1. *If algorithm* $\boldsymbol{Lift^R}$ *FAILS on a Monotone CNF query* Q*, then computing* $\Pr(Q)$ *is #P-hard.*

However, the inclusion of negations in our query language increases significantly the difficulty of analyzing query complexities. Our major technical result extends Theorem 1 to a class of CNF queries with negation.

Define a *Type-1* query to be a CNF formula where each clause has at most two variables denoted x, y, and each atom is of one of the following three kinds:

- Unary symbols $R_1(x), R_2(x), R_3(x), \ldots$
- Binary symbols $S_1(x,y), S_2(x,y), \ldots$
- Unary symbols $T_1(y), T_2(y), \ldots$

or the negation of these symbols.

Our main result is:

THEOREM 2. *For every Type-1 query* Q*, if algorithm* $\boldsymbol{Lift^R}$ *FAILS then computing* $\Pr(Q)$ *is #P-hard.*

The proof is a significant extension of the techniques used by [1] to prove Theorem 1. The primary difficulty in extending the hardness results to queries with negation comes from the failure of the *zigzag construction*. Given a query Q, the zigzag construction first builds a PDB with domain equal to the union of distinct sets $U^0, U^1, U^2, V^0, V^1, V^2$. Next, the zigzag construction encodes the probabilities of each relation such that $\Pr(Q) = \Pr(Q')$, where $Q' = Q_1[U^0, V^1] \wedge Q_2[U^1, V^1] \wedge Q_3[U^1, V^2] \wedge Q_4[U^2, V^2] \wedge Q_5[U^2, V^0]$. Each $Q_i[A, B]$ is a copy of Q quantified only over $x \in A, y \in B$ and with distinct relational atoms, except for designated unary atoms, which connect Q in a linear chain from $Q_1 \rightarrow Q_2 \rightarrow \cdots$. However, for the equivalence between Q and Q' to hold, we must ensure that $\Pr(Q)$ does not depend on any clause with one variable in, for instance, U^0, and another variable in V^2 (which is a clause not included in Q').

This is simple in the monotone case: set such unwanted tuples to have probability one, and these components of the query disappear. If the same relation appears negated elsewhere in the query, this is no longer guaranteed to work.

This obstacle required developing an entirely new reduction from #PP2CNF [5] for Type-1 queries Q where our algorithm reports failure. The reduction consists of a combinatorial part (the construction of certain gadgets), and an algebraic part, which makes novel use of the concepts of algebraic independence [10] and annihilating polynomials [4]. We prove that, for any Q where $\mathbf{Lift^R}$ reports failure, the reduction encodes an invertible system of equations such that computing $\Pr(Q)$ in PTIME enables computing the solution to any instance of the #P-hard problem #PP2CNF in PTIME. Proving the system is invertible is further complicated by the loss of the zigzag construction, as we no longer have a Kronecker product of Vandermonde matrices, requiring a more complicated analysis to prove the resulting matrix is non-singular. Details of this and the rest of the proof appear in the full paper [3].

4. FUTURE DIRECTIONS

Theorem 2 represents the first step towards proving the following conjecture: $\mathbf{Lift^R}$ provides a dichotomy for probabilistic queries on arbitrary universally-quantified CNF sentences where negation can be applied anywhere. Future work aims to prove this conjecture by expanding the existing dichotomy beyond Type-1 queries. We are also exploring new types of queries, and different assumptions about the probabilistic database, in particular symmetric probabilities, and how they affect our algorithm and dichotomy.

5. ACKNOWLEDGMENTS

This work was partially supported by ONR grant #N00014-12-1-0423, NSF grants IIS-1115188 and IIS-1118122, and the Research Foundation-Flanders (FWO-Vlaanderen).

6. REFERENCES

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

[2] V. Gogate and P. Domingos. Probabilistic theorem proving. In *Proceedings of UAI*, pages 256–265, 2011.

[3] E. Gribkoff, G. Van den Broeck, and D. Suciu. Understanding the Complexity of Lifted Inference and Asymmetric Weighted Model Counting. *ArXiv e-prints*, May 2014. arXiv:1405.3250 [cs.AI].

[4] N. Kayal. The complexity of the annihilating polynomial. In *Computational Complexity, 2009. CCC'09. 24th Annual IEEE Conference on*, pages 184–193. IEEE, 2009.

[5] J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4):777–788, 1983.

[6] D. Suciu, D. Olteanu, C. Ré, and C. Koch. Probabilistic databases. *Synthesis Lectures on Data Management*, 3(2):1–180, 2011.

[7] G. Van den Broeck. *Lifted Inference and Learning in Statistical Relational Models*. PhD thesis, KU Leuven, 2013.

[8] G. Van den Broeck, W. Meert, and A. Darwiche. Skolemization for weighted first-order model counting. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2014.

[9] G. Van den Broeck, N. Taghipour, W. Meert, J. Davis, and L. De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In *IJCAI*, pages 2178–2185, 2011.

[10] J.-T. Yu. On relations between jacobians and minimal polynomials. *Linear algebra and its applications*, 221:19–29, 1995.