Discovering Dichotomies for Problems in Database Theory

Neha Makhija

Supervised by Wolfgang Gatterbauer.
Northeastern University, 360 Huntington Ave, Boston Massachusetts 02118, USA.

Abstract
Dichotomy theorems, which characterize the conditions under which a problem can be solved efficiently, have helped identify important tractability borders for probabilistic query evaluation, view maintenance, query containment (among many more problems). However, dichotomy theorems for many such problems remain elusive under key settings such as bag semantics or for queries with self-joins. This work aims to unearth dichotomies for fundamental problems in reverse data management and knowledge representation. We use a novel approach to discovering dichotomies: instead of creating dedicated algorithms for easy (PTIME) and hard cases (NP-complete), we devise *unified algorithms* that are guaranteed to terminate in PTIME for easy cases. Using this approach, we discovered new tractable cases for the problem of minimal factorization of provenance formulas as well as dichotomies under bag semantics for the problems of resilience and causal responsibility.

Keywords
Reverse Data Management, Factorization, Resilience, Causal Responsibility, Dichotomy, Bag Semantics, Self-Join

1. Introduction

Our goal is to understand the complexity of solving three distinct, yet related, problems: resilience, causal responsibility and minimal factorization. They underlie many practical problems such as reverse data management [3], (including view maintenance [4, 5, 6], explanations and diagnostics [7, 8, 9]), knowledge representation as boolean formulas [10], and probabilistic inference [11].

We treat all problems with the same novel method: Rather than deriving a dedicated PTIME algorithm for certain queries (and proving hardness for the rest), we propose a unified Integer Linear Program (ILP) formulation for *all* conjunctive queries under *all* problem variants. We then show that, for all PTIME queries, *the Linear Program (LP) relaxation of our ILP has the same optimal value*, proving that existing ILP solvers are *guaranteed* to solve problems for those queries in PTIME. Through this method, we are able to uncover new tractable cases and show the first dichotomy under bag semantics in this space [2] (Fig. 1). In addition, we provide a unified hardness criterion by proving a prior conjecture [12] that states that if a query can form an "Independent Join Path", then it must be hard [2]. This unified hardness criterion helps us prove the hardness of some queries with self-joins, whose complexities were previously unknown.

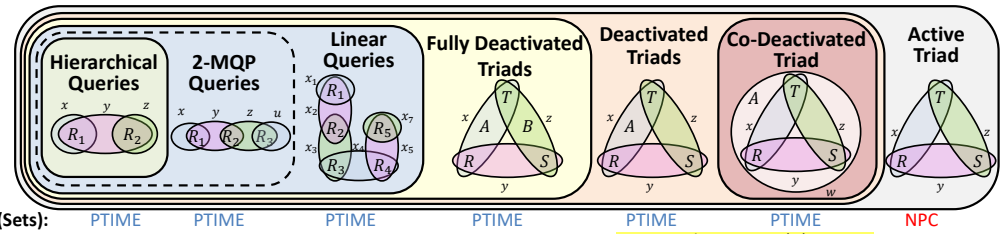
Resilience. What is the minimal number of tuples to delete from a database to eliminate all query answers?

Problem 1 (Resilience). *How surprising is it for an Oscar winning actor to act in a movie directed by their spouse? We can quantify this by calculating the resilience of the query* $Q_A^{\triangle}$:− *Oscar(actor), ActsIn(actor, movie), DirectedBy(movie, dir), Spouse(actor, dir). Resilience does not equate to simply the number of output rows but rather asks for the minimum number of changes in the world needed to have no satisfying output. For example, if we do not include the spouse pair of Frances McDormand and Joel Coen, the single deletion would take away 9 rows from the output. Intuitively, if the resilience is small, there have been a small number of events that have led to an Oscar winning actor being in a movie directed by their spouse.*

Interestingly, the resilience for this query can be calculated in PTIME *under set semantics, but not bag semantics (such as when accounting for multiple Oscar wins). If we now change the query to remove the constraint of the actor having won an Oscar, then finding the resilience of the resulting query* $Q^{\triangle}$:− *ActsIn(actor, movie), DirectedBy(movie, dir), Spouse(actor, dir) is* NPC*!*

Resilience was introduced [13] to capture the essence of all reverse data management questions: What is a minimum set of changes to a database to produce a certain change in the output of a query? The same work gave a dichotomy [13] of the complexity for resilience for self-join free queries under set semantics. While a dichotomy for the general self-join case remains open, Freire et al. [12] gave partial complexity results in this space, and conjectured that the notion of Independent Join Paths (IJPs) is a sufficient and necessary condition for hardness of resilience. We proved this conjecture for self-join free queries (with a slight fix of the original statement) [2], and proved that IJPs are a sufficient condition for hardness in queries with self-joins as well.

VLDB 2023 PhD Workshop, co-located with the 49th International Conference on Very Large Data Bases (VLDB 2023), August 28, 2023, Vancouver, Canada
makhija.n@northeastern.edu (N. Makhija)
nehamakhija.github.io (N. Makhija)
0000-0003-0221-6836 (N. Makhija)



	Hierarchical Queries	2-MQP Queries	Linear Queries	Fully Deactivated Triads	Deactivated Triads	Co-Deactivated Triad	Active Triad
RES(Sets):	PTIME	PTIME	PTIME	PTIME	PTIME	PTIME	NPC
RSP(Sets):	PTIME	PTIME	PTIME	PTIME	PTIME: dominating (A) NPC: dominated (R, S, T)		NPC
RES(Bags):	PTIME	PTIME	PTIME	NPC	NPC	NPC	NPC
RSP(Bags):	PTIME	PTIME	PTIME	NPC	NPC	NPC	NPC
FACT(Sets):	PTIME	PTIME	?	?	?	NPC	NPC
PROB(Sets):	PTIME	#P-hard	#P-hard	#P-hard	#P-hard	#P-hard	#P-hard

Figure 1: Overview of complexity results for all self-join free conjunctive queries. The space of all queries is broken down into classes with different complexities as defined in the full papers [1, 2]. Results highlighted with a yellow background are new. RES denotes resilience, RSP causal responsibility, FACT minimal factorization, while PROB denotes probabilistic query evaluation.

Causal Responsibility. For a given tuple in the output, what is a minimum subset of input tuples to remove to make the given tuple "counterfactual"?

Problem 2 (Causal Responsibility). *Assume we wished to ask: "What is the responsibility of Frances McDormand's Oscar win towards the output of our query?" If this Oscar was solely responsible for the output, it would be a counterfactual cause – i.e. if she had not won, there would be no satisfying output. However, this tuple still has "partial" responsibility. By measuring how far we are from a world where the tuple is counterfactual, we can get a notion of its responsibility to the output.*[a] *Interestingly, due to our new fine-grained complexity results, we can find the responsibility of a particular Oscar win in* PTIME*, but finding the responsibility of a tuple from the ActsIn, DirectedBy or Spouse Table is* NPC*.*

[a]The responsibility is inversely proportional to the minimum number of tuples to be deleted $|\tau|$ and is given by $1/(1+|\tau|)$.

This notion of causal responsibility is due to foundational work by Halpern, Pearl, et al. [14] which defined causal responsibility based on *minimal interventions* in the causal graph. Meliou et al. [15] adapted this concept to define causal responsibility for database queries and showed a dichotomy under set semantics for self-join free queries. Our work proves a dichotomy under bag semantics, as well as more fine-grained result based on the relation of the tuple we find responsibility of [2].

Minimal Factorization. Given the provenance formula for a Boolean query, what is its *minimal size equivalent expression*?

Problem 3 (Minimal Factorization). *For the same query $Q_A^{\triangle}$, how could we representing its output efficiently? Imagine the output was due to a single Oscar winning actor o_1, who was married once represented by s_1, and acted in two movies a_1, a_2, while their spouse also directed these same movies according to tuples d_1, d_2. We could then represent the output of this query with the boolean provenance formula $o_1s_1a_1d_1 + o_1s_1a_2d_2$. However, notice that a factorized representation $o_1s_1(a_1d_1 + a_2d_2)$ is also equivalent (and minimal in this case). We showed that the minimal factorization for $Q_A^{\triangle}$ can be found in* PTIME *[1], however the complexity of self-join free queries remains open in general.*

Minimal factorization solves the Minimum Equivalent Expression (MEE) [16, 10] problem for provenance formulas. Previously work [17] on this problem has lead to work on factorized databases [18]. However, the dichotomy for finding the exact minimal factorization is open. Minimal factorizations of provenance can be used to obtain probabilistic inference bounds. Prior approaches for approximate probabilistic inference are either incomplete i.e. focus on just PTIME cases [11, 19, 20], or do not solve all PTIME cases exactly [11, 21]. Using minimal factorization achieves the best of both worlds i.e. it applies to easy and hard cases while recovering all known PTIME cases exactly. While we do not yet prove a dichotomy for minimal factorization, the quest for a dichotomy has helped prove a large tractable class: queries with 2 *Minimal Query Plans* (2-MQP).[1] We also place the set of tractable queries for minimal factorization between resilience under set semantics and probabilistic inference in the self-join free case (Fig. 1)

2. Unified ILP-Based Framework

For all three problems, we can construct ILPs whose size (number of constraints and variables) is polynomial in

[1]We hypothesize that 2-MQP Queries are a subset of linear queries.

the size of the database instance. While solving ILPs is NPC in general, we proved that the PTIME relaxations of our programs are correct for all easy cases, thus creating a unified framework that solves all known easy cases in PTIME. We now provide intuition for these ILPs. For more details of construction and examples, please see the full papers [1, 2].

Resilience. Intuitively, each tuple in the input is associated with a binary decision variable, which is set to 1 if the tuple should be deleted else set to 0. The objective of the ILP is to minimize the number of tuples, which is the same as minimizing the sum of the variables. For bag semantics, one can simply use a weighted sum, where the weight is the number of copies of a tuple in the database. We have a constraint for each output row - that it must be deleted. An output row is deleted if any of the tuples contributing to it are deleted. Thus, for each output row, the sum of the variables of the tuple in the row must be ≥ 1 i.e. at least one of the tuples must be deleted.

Causal Responsibility. To make a tuple counterfactual, one must delete all output rows it does not contribute to (which is identical to the resilience problem), but with the additional constraint that *some output row must be preserved.* Thus, the causal responsibility ILP has the variables and constraints of the resilience ILP, plus additional variables to indicate if an output row is deleted, as well as one additional counterfactual constraint that enforces that all output rows cannot be deleted.

Minimal Factorization. The ILP for Minimal Factorization is based on the idea that different factorizations of a provenance formulas are equivalent to evaluating different output rows with different query plans. Thus, the ILP assigns a query plan to each output row.[2] The objective, equal to the length of the factorized expression, is calculated by summing up projections from the different query plans a tuple may be used in.

3. Key Results

Result 1: ILP Relaxations recover all known PTIME cases. We prove that for all prior known and newly found PTIME cases for all three problems, our ILPs are solved in guaranteed PTIME by standard solvers. For the problems of resilience [2] and minimal factorization [1], we use the LP relaxation, i.e. all integrality constraints on all decision variables are removed. However, for causal responsibility, we relax all variables except those that track if an output row is preserved, creating a Mixed Integer Linear Program (MILP). While MILPs take exponential time in general, we show that the proposed MILP relaxation for causal responsibility can be solved in PTIME if the query is known to be in PTIME [2].

[2]We show that considering only the "*minimal query plans*" suffices to obtain the minimal factorization.

Result 2: Dichotomies under Bag Semantics. Real-world databases consist of bags instead of sets i.e. a relation may have multiple copies of the same tuple. However, the complexity of the many problems under bag semantics is not well understood. We show that both resilience and responsibility are easy under bag semantics if and only if the query is *linear* [2]. Notice that the tractability frontier for bag semantics differs from set semantics, where responsibility is a strictly harder problem than resilience (Fig. 1). We show that switching from set semantics to bag semantics need only change the objective function of the ILP, and the constraint matrix remains the same.

Result 3: New Tractable Cases. We are able to show that for any query with ≤ 2 minimal query plans, minimal factorization can be solved in PTIME as the relaxation of the minimal factorization ILP is always correct [1]. This large class of queries includes hierarchical queries as a special case. In addition, a fine-grained analysis of the complexity of causal responsibility based on the relation of the tuple we find the responsibility of, reveals additional tractable cases (Fig. 1).

Result 4: Instance-Based Tractability. We prove cases such that our unified algorithm is *guaranteed to terminate in* PTIME, even for hard queries. For example, we show that when the provenance of a query output is read-once [20], all three problems can be solved in PTIME. The interesting aspect is that our unified algorithm *does not need to know about these conditions as input*, it just automatically leverages those during query time.

Result 5: Unified Hardness Criterion. Freire et al. [12] conjectured that the ability to construct Independent Join Paths (IJPs) is a necessary and sufficient criterion to prove hardness of resilience for a query. We proved this conjecture for the self-join free case, and the sufficiency criterion for all cases[3], and use it as a unified way to show hardness for all three problems. With this unified hardness criterion, we obtain considerably simplified proofs, and prove hardness for a query with self-join whose complexity was previously unknown [2]. We give a Disjunctive Logic Program (DLP) formulation that can computationally derive such hardness certificates, and thus has shown several queries whose complexity was previously unknown to be hard. We also show, with the existence of an IJP for query $A(a), R(a,x,y), S(a,y,z), T(a,z,x)$ that the tractable cases for minimal factorization are a strict subset of the tractable cases for resilience under set semantics.

4. Conclusion and Future Work

This overview gives the intuition for a novel way of determining the complexity of problems such as resilience,

[3]With a slight generalization of the definition of IJP [2]

causal responsibility and minimal factorization. We give a universal encoding as ILP and show that a relaxation is guaranteed to give an integral solution for all PTIME cases, thereby proving that modern solvers can return the answer in guaranteed PTIME. While this approach is known in the optimization literature [22], it has so far not been applied as *proof method* to establish dichotomy results in reverse data management to the best of our knowledge. Since the resulting theory is simpler and naturally captures all prior known PTIME cases, we believe that this approach will also help in related open problems in data management, in particular a so far elusive complete dichotomy for resilience of queries with self-joins [12] and completing the dichotomy for minimal factorization. The techniques in this paper are not limited to the three problems we consider, but may be helpful to discover other dichotomies as well.

References

[1] N. Makhija, W. Gatterbauer, Towards a dichotomy for minimally factorizing the provenance of self-join free conjunctive queries, arXiv preprint arXiv:2105.14307 (2021). doi:10.48550/arXiv.2105.14307.

[2] N. Makhija, W. Gatterbauer, A unified approach for resilience and causal responsibility with integer linear programming (ilp) and lp relaxations, arXiv preprint arXiv:2212.08898 (2022). doi:10.48550/arXiv.2212.08898.

[3] A. Meliou, W. Gatterbauer, D. Suciu, Reverse data management, PVLDB 4 (2011) 1490–1493. doi:10.14778/3402755.3402803.

[4] P. Buneman, S. Khanna, W.-C. Tan, On propagation of deletions and annotations through views, in: PODS, 2002, pp. 150–158. doi:10.1145/543613.543633.

[5] U. Dayal, P. A. Bernstein, On the correct translation of update operations on relational views, ACM TODS 7 (1982) 381–416. doi:10.1145/319732.319740.

[6] X. Hu, S. Sun, S. Patwa, D. Panigrahi, S. Roy, Aggregated deletion propagation for counting conjunctive query answers, PVLDB 14 (2020) 228–240. doi:10.14778/3425879.3425892.

[7] S. Roy, D. Suciu, A formal approach to finding explanations for database queries, in: SIGMOD, 2014, pp. 1579–1590. doi:10.1145/2588555.2588578.

[8] B. Glavic, A. Meliou, S. Roy, Trends in explanations: Understanding and debugging data-driven systems, Foundations and Trends in Databases 11 (2021). doi:10.1561/9781680838817.

[9] X. Wang, A. Meliou, E. Wu, Qfix: Diagnosing errors through query histories, in: Proceedings of the 2017 ACM International Conference on Management of Data, 2017, pp. 1369–1384. doi:10.1145/3035918.3035925.

[10] D. Buchfuhrer, C. Umans, The complexity of boolean formula minimization, Journal of Computer and System Sciences 77 (2011) 142–153. doi:10.1016/j.jcss.2010.06.011.

[11] N. N. Dalvi, D. Suciu, Efficient query evaluation on probabilistic databases, VLDB J. 16 (2007) 523–544. doi:10.1007/s00778-006-0004-3.

[12] C. Freire, W. Gatterbauer, N. Immerman, A. Meliou, New results for the complexity of resilience for binary conjunctive queries with self-joins, in: PODS, 2020, pp. 271–284. doi:10.1145/3375395.3387647.

[13] C. Freire, W. Gatterbauer, N. Immerman, A. Meliou, The complexity of resilience and responsibility for self-join-free conjunctive queries, PVLDB 9 (2015) 180–191. doi:10.14778/2850583.2850592.

[14] J. Y. Halpern, J. Pearl, Causes and explanations: A structural-model approach. Part I: Causes, Brit. J. Phil. Sci. 56 (2005) 843–887. doi:10.1093/bjps/axi147.

[15] A. Meliou, W. Gatterbauer, K. F. Moore, D. Suciu, The complexity of causality and responsibility for query answers and non-answers, PVLDB 4 (2010) 34–45. doi:10.14778/1880172.1880176.

[16] M. R. Garey, D. S. Johnson, Computers and intractability, volume 174, W. H. Freeman & Co., 1979. doi:10.5555/578533.

[17] D. Olteanu, J. Závodnỳ, Factorised representations of query results: size bounds and readability, in: Proceedings of the 15th International Conference on Database Theory, 2012, pp. 285–298. doi:10.1145/2274576.2274607.

[18] D. Olteanu, M. Schleich, Factorized databases, SIGMOD Rec. 45 (2016) 5–16. doi:10.1145/3003665.3003667.

[19] S. Roy, V. Perduca, V. Tannen, Faster query answering in probabilistic databases using read-once functions, in: ICDT, 2011, pp. 232–243. doi:10.1145/1938551.1938582.

[20] P. Sen, A. Deshpande, L. Getoor, Read-once functions and query evaluation in probabilistic databases, PVLDB 3 (2010) 1068–1079. doi:10.14778/1920841.1920975.

[21] W. Gatterbauer, D. Suciu, Dissociation and propagation for approximate lifted inference with standard relational database management systems, VLDB J. 26 (2017) 5–30. doi:10.1007/s00778-016-0434-5.

[22] A. Schrijver, Combinatorial optimization: polyhedra and efficiency, volume 24 of *Algorithms and Combinatorics*, Springer, 2003. URL: https://link.springer.com/book/9783540443896.