Counting Answers to Unions of Conjunctive Queries: Natural Tractability Criteria and Meta-Complexity

Anonymous Author(s)***

ABSTRACT

We study the problem of counting answers to unions of conjunctive queries (UCQs) under structural restrictions on the input query. Concretely, given a class C of UCQs, the problem #UCQ(C) provides as input a UCQ $\Psi \in C$ and a database $\mathcal{D}$ and the problem is to compute the number of answers of Ψ in $\mathcal{D}$.

Chen and Mengel [PODS'16] have shown that for any recursively enumerable class C, the problem #UCQ(C) is either fixed-parameter tractable or hard for one of the parameterised complexity classes W[1] or #W[1]. However, their tractability criterion is unwieldy in the sense that, given any concrete class C of UCQs, it is not easy to determine how hard it is to count answers to queries in C. Moreover, given a single specific UCQ Ψ, it is not easy to determine how hard it is to count answers to Ψ.

In this work, we address the question of finding a *natural* tractability criterion: The combined conjunctive query of a UCQ $\Psi = \varphi_1 \vee \cdots \vee \varphi_\ell$ is the conjunctive query $\bigwedge(\Psi) = \varphi_1 \wedge \cdots \wedge \varphi_\ell$. We show that under natural closure properties of C, the problem #UCQ(C) is fixed-parameter tractable if and only if the combined conjunctive queries of UCQs in C, and their contracts, have bounded treewidth. A contract of a conjunctive query is an augmented structure, taking into account how the quantified variables are connected to the free variables — if all variables are free, then a conjunctive query is equal to its contract; in this special case the criterion for fixed-parameter tractability of #UCQ(C) thus simplifies to the combined queries having bounded treewidth.

Finally, we give evidence that a closure property on C is necessary for obtaining a natural tractability criterion: We show that even for a single UCQ Ψ, the meta problem of deciding whether #UCQ($\{\Psi\}$) can be solved in time $O(|\mathcal{D}|^d)$ is NP-hard for any fixed $d \geq 1$. Moreover, we prove that a known exponential-time algorithm for solving the meta problem is optimal under assumptions from fine-grained complexity theory. As a corollary of our reduction, we also establish that approximating the Weisfeiler-Leman-Dimension of a UCQ is NP-hard.

CCS CONCEPTS

• **Theory of computation → Design and analysis of algorithms**; • **Information systems → Relational database query languages**.

KEYWORDS

counting problems, conjunctive queries, unions of conjunctive queries, simplicial complexes, Weisfeiler-Leman algorithm

ACM Reference Format:

Anonymous Author(s). 2022. Counting Answers to Unions of Conjunctive Queries: Natural Tractability Criteria and Meta-Complexity. In *Proceedings of the 43rd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '24), June 9–15, 2024, Santiago, Chile.* ACM, New York, NY, USA, 23 pages. https://doi.org/10.1145/3517804.3526231

1 INTRODUCTION

Conjunctive queries are among the most fundamental and well-studied objects in database theory [2, 21, 47, 48, 60, 61]. A conjunctive query (CQ) φ with free variables $X = \{x_1, \ldots, x_k\}$ and quantified variables $Y = \{y_1, \ldots, y_d\}$ is of the form

$$\varphi(X) = \exists Y\, R_1(\mathbf{t}_1) \wedge \ldots \wedge R_n(\mathbf{t}_n),$$

where $R_1, \ldots, R_n$ are relational symbols and each $\mathbf{t}_i$ is a tuple of variables from $X \cup Y$. A database $\mathcal{D}$ consists of a set of elements $U(\mathcal{D})$, denoted the *universe* of $\mathcal{D}$, and a set of relations over this universe. The corresponding relation symbols are the *signature* of $\mathcal{D}$. If $R_1, \ldots, R_n$ are in the signature of $\mathcal{D}$ then an *answer* of φ in $\mathcal{D}$ is an assignment $a\colon X \to U(\mathcal{D})$ that has an extension to the existentially quantified variables Y that agrees with all the relations $R_1, \ldots, R_n$. Even more expressive is a union of conjunctive queries (UCQ). Such a union is of the form $\Psi(X) = \varphi_1(X) \vee \ldots \vee \varphi_\ell(X)$, where each $\varphi_i(X)$ is a CQ with free variables X. An answer to Ψ is then any assignment that is answer to at least one of the CQs in the union.

Since evaluating a given CQ on a given database is NP-complete [21] a lot of research focused on finding tractable classes of CQs. A fundamental result by Grohe, Schwentick, and Segoufin [42] established that the tractability of evaluating all CQs of bounded arity whose Gaifman graph is in some class of graphs C depends on whether or not the treewidth in C is bounded.

More generally, finding an answer to a conjunctive query can be cast as finding a (partial) homomorphism between relational structures, and therefore is closely related to the framework of constraint satisfaction problems. In this setting, Grohe [40] showed that treewidth modulo homomorphic equivalence is the right criterion for tractability. There is also an important line of work [38, 41, 50] culminating in the fundamental work by Marx [51] that investigates the parameterised complexity for classes of queries with unbounded arity. In general, tractability of conjunctive queries is closely related to how "tree-like" or close to acyclic they are.

Counting answers to CQs has also received significant attention in the past [3, 23, 29, 31, 32, 39, 56]. Chen and Mengel [22] gave a complete classification for the counting problem on classes of CQs (with bounded arity) in terms of a natural criterion loosely based on treewidth. They present a trichotomy into fixed-parameter

PODS '24, June 9–15, 2024, Santiago, Chile

ACM ISBN 978-1-4503-9260-0/22/06…$15.00
https://doi.org/10.1145/3517804.3526231

tractable, W[1]-complete, and #W[1]-complete cases. In subsequent work [23], this classification was extended to unions of conjunctive queries (and to even more general queries in [31]). However, for UCQs, the established criteria for tractability and intractability are implicit (see [23, Theorems 3.1 and 3.2]) in the sense that, given a specific UCQ Ψ, it is not at all clear how hard it is to count answers to Ψ based on the criteria in [23]. To make this more precise: It is not even clear whether we can, in polynomial time in the size of Ψ, determine whether answers to Ψ can be counted in linear time in the input database.

1.1 Our contributions

With the goal of establishing a more practical tractability criterion for counting answers to UCQs, we explore the following two main questions in this work:

> Q1) Is there a natural criterion that captures the fixed-parameter tractability of counting answers to a class of UCQs, parameterised by the size of the query?
>
> Q2) Is there a natural criterion that captures whether counting answers to a single fixed UCQ is linear-time solvable (in the size of a given database)?

Question Q1): Fixed-Parameter Tractability. For a class C of UCQs, we consider the problem #UCQ(C) that takes as input a UCQ Ψ from C and a database $\mathcal{D}$, and asks for the number $\mathsf{ans}(\Psi \to \mathcal{D})$ of answers of Ψ in $\mathcal{D}$. We assume that the arity of the UCQs in C is bounded, that is, there is constant c such that each relation that appears in some query in C has arity at most c. As explained earlier, due to a result of Chen and Mengel [23], there is a known but rather unwieldy tractability criterion for #UCQ(C), when the problem is parameterised by the size of the query. On a high level, the number of answers of a UCQ Ψ in a given database can be expressed as a finite linear combination of CQ answer counts, using the principle of inclusion-exclusion. This means that $\mathsf{ans}(\Psi \to \mathcal{D})$ is equal to $\sum_i c_i \cdot \mathsf{ans}(\varphi_i \to \mathcal{D})$, where each φ_i is simply a conjunctive query (and not a union thereof). We refer to this linear combination as the *CQ expansion* of Ψ. Chen and Mengel showed that the parameterised complexity of computing $\mathsf{ans}(\Psi \to \mathcal{D})$ is guided by the hardest term in the respective CQ expansion. The complexity of computing these terms is simply the complexity of counting the answers of a conjunctive query, and this is well understood [22]. Hence, the main challenge for this approach is to understand the linear combination, i.e., to understand for which CQs the corresponding coefficients are non-zero. The problem is that the coefficients c_i of these linear combinations are alternating sums, which in similar settings have been observed to encode algebraic and even topological invariants [57]. This makes it highly non-trivial to determine which CQs actually contribute to the linear combination. We introduce the concepts required to state this classification informally, the corresponding definitions are given in Section 2.

We first give more details about the result of [22]. Let $\Gamma(C)$ be the class of those conjunctive queries that contribute to the CQ expansion of at least one UCQ in C, and that additionally are what we call #minimal. Intuitively, a conjunctive query φ is #minimal if there is no proper subquery φ' of φ that has the same number of answers as φ in every given database. Then the tractability criterion depends on the treewidth of the CQs in $\Gamma(C)$. It also depends on the treewidth of the corresponding class $\mathsf{contract}(\Gamma(C))$ of contracts (formally defined in Definition 17), which is an upper bound of what is called the "star size" in [32] and the "dominating star size" in [31]. Here is the formal statement of the known dichotomy for #UCQ(C).

Theorem 1 ([23])**.** *Let C be a recursively enumerable class of UCQs of bounded arity. If the treewidth of $\Gamma(C)$ and of $\mathsf{contract}(\Gamma(C))$ is bounded, then #UCQ(C) is fixed-parameter tractable. Otherwise, #UCQ(C) is W[1]-hard.*

We investigate under which conditions this dichotomy can be simplified. We show that for large classes of UCQs there is actually a much more natural tractability criterion that does not rely on $\Gamma(C)$, i.e., here the computation of the coefficients of the linear combinations as well as the concept of #minimality do not play a role. We first show a simpler classification for UCQs without existential quantifiers. To state the results we require some additional definitions: The *combined query* $\bigwedge(\Psi)$ of a UCQ $\Psi(X) = \varphi_1(X) \vee \cdots \vee \varphi_\ell(X)$ is the conjunctive query obtained from Ψ by replacing each disjunction by a conjunction, that is $\bigwedge(\Psi) = \varphi_1(X) \wedge \cdots \wedge \varphi_\ell(X)$. Given a class of UCQs C, we set $\bigwedge(C) = \{\bigwedge(\Psi) \mid \Psi \in C\}$.

It will turn out that the structure of the class of combined queries $\bigwedge(C)$ determines the complexity of counting answers to UCQs in C, given that C has the following natural closure property: We say that C is *closed under deletions* if, for all $\Psi(X) = \varphi_1(X) \vee \cdots \vee \varphi_\ell(X)$ and for every $J \subseteq [\ell]$, the subquery $\bigvee_{j \in J} \varphi_j(X)$ is also contained in C. For example, any class of UCQs defined solely by the conjunctive queries admissible in the unions (such as unions of acyclic conjunctive queries) is closed under deletions. The following classification resolves the complexity of counting answers to UCQs in classes that are closed under deletions; we will see later that the closedness condition is necessary. Moreover, the tractability criterion depends solely on the structure of the combined query, and not on the terms in the CQ expansion, thus yielding, as desired, a much more concise and natural characterisation. As mentioned earlier, we first state the classification for quantifier-free UCQs.

Theorem 2. *Let C be recursively enumerable class of quantifier-free UCQs of bounded arity. If $\bigwedge(C)$ has bounded treewidth then #UCQ(C) is fixed-parameter tractable. If $\bigwedge(C)$ has unbounded treewidth and C is closed under deletions then #UCQ(C) is W[1]-hard.*

We emphasise here that Theorem 2 is in terms of the simpler object $\bigwedge(C)$ instead of the complicated object $\Gamma(C)$.

If we allow UCQs with quantified variables in the class C then the situation becomes more intricate. Looking for a simple tractability criterion that describes the complexity of #UCQ(C) solely in terms of $\bigwedge(C)$ requires some additional effort. First, for a UCQ Ψ that has quantified variables, $\mathsf{contract}(\Psi)$ is not necessarily the same as Ψ, and therefore the treewidth of the contracts also plays a role. Moreover, the matching lower bound requires some conditions in addition to being closed under deletions. Nevertheless, our result is in terms of the simpler objects $\bigwedge(C)$ and $\mathsf{contract}(\bigwedge(C))$ rather than the more complicated $\Gamma(C)$ and $\mathsf{contract}(\Gamma(C))$. For Theorem 3, recall that a conjunctive query is *self-join-free* if each relation symbol occurs in at most one atom of the query.

Theorem 3. *Let C be a recursively enumerable class of UCQs of bounded arity. If $\wedge(C)$ and $\mathsf{contract}(\wedge(C))$ have bounded treewidth then #UCQ(C) is fixed-parameter tractable. Otherwise, if (I)–(III) are satisfed, then #UCQ(C) is* W[1]*-hard.*

(I) C is closed under deletions.
(II) The number of existentially quantified variables of queries in C is bounded.
(III) The UCQs in C are unions of self-join-free conjunctive queries.

In Appendix E we show that Theorem 3 is tight in the sense that, if any of these conditions is dropped, there are counterexamples to the claim that tractability is guided solely by $\wedge(C)$ and $\mathsf{contract}(\wedge(C))$.

Question Q2): Linear-Time Solvability. Now we turn to the question of linear-time solvability for a single fixed UCQ. The huge size of databases in modern applications motivates the question of which query problems are actually linear-time solvable. Along these lines, there is a lot of research for enumeration problems [8, 11, 12, 15, 20, 61].

The question whether counting answers to a conjunctive query φ can be achieved in time linear in the given database has been studied previously [52]. The corresponding dichotomy is well known and was discovered multiple times by different authors in different contexts.[1] In these results, the tractability criterion is whether φ is *acyclic*, i.e., whether it has a join tree (see [37]). The corresponding lower bounds are conditioned on a widely used complexity assumption from fine-grained complexity, namely the Triangle Conjecture. We define all of the complexity assumptions that we use in this work in Section 2. There we also formally define the size of a database (as the sum of the size of its signature, its universe, and its relations).

It is well-known that counting answers to quantifier-free conjunctive queries can be done in linear time if and only if the query is acyclic. The "only if" part relies on hardness assumptions from fine-grained complexity theory. Concretely, we have

Theorem 4 (See Theorem 12 in [17], and [7, 8, 11])**.** *Let φ be a quantifier-free conjunctive query and suppose that the Triangle Conjecture is true. Then the number of answers of φ in a given database $\mathcal{D}$ can be computed in time linear in the size of $\mathcal{D}$ if and only if φ is acyclic.*

We note that the previous theorem is false if quantified variables were allowed as this would require the consideration of semantic acyclicity[2] (see [10]).

Theorem 4 yields an efficient way to check whether counting answers to a quantifier-free conjunctive query φ can be done in linear time: Just check whether φ is acyclic (in polynomial time, see for instance [37]). We investigate the corresponding question for *unions* of conjunctive queries. In stark contrast to Theorem 4, we show that there is no efficiently computable criterion that determines the linear-time tractability of counting answers to *unions* of conjunctive queries, unless some conjectures of fine-grained complexity theory fail.

We first observe that, as in the investigation of question Q1), one can obtain a criterion for linear-time solvability by expressing UCQ answer counts as linear combinations of CQ answer counts. Concretely, by a straightforward extension of previous results, we show that, assuming the Triangle Conjecture, a linear combination of CQ answer counts can be computed in linear time if and only if the answers to each #minimal CQ in the linear combination can be computed in linear time, that is, if each such CQ is acyclic. However, this criterion is again unwieldy in the sense that, for all we know, it may take time exponential in the size of the respective UCQ to determine whether this criterion holds.

In view of our results for question Q1) about fixed-parameter tractability, one might suspect that a more natural and simpler tractability criterion exists. However, it turns out that even under strong restrictions on the UCQs that we consider, an efficiently computable criterion is unlikely. We make this formal by studying the following meta problem.

Name: META
Input: A union Ψ of quantifier-free conjunctive queries.
Output: Is it possible to count answers to Ψ in time linear in the size of $\mathcal{D}$.

Restricting the input of META to quantifier-free queries is sensible as, without this restriction, the meta problem is known to be NP-hard even for conjunctive queries: If all variables are existentially quantified, then evaluating a conjunctive query can be done in linear time if and only if the query is semantically acyclic [61] (the "only if" relies on standard hardness assumptions). However, verifying whether a conjunctive query is semantically acyclic is already NP-hard [10]. In contrast, when restricted to quantifier-free conjunctive queries, the problem META is polynomially-time solvable according to Theorem 4.

We can now state our main result about the complexity of META. The hardness results hold under substantial additional input restrictions, which make these results stronger.

Theorem 5. *META can be solved in time $2^{O(\ell)} \cdot |\Psi|^{\mathrm{poly}(\log|\Psi|)}$, where ℓ is the number of conjunctive queries in the union, if the Triangle Conjecture is true. Moreover,*

- *If the Triangle Conjecture is true then META is* NP*-hard. If, additionally, ETH is true, then META cannot be solved in time $2^{o(\ell)}$.*
- *If SETH is true then META is* NP*-hard and cannot be solved in time $2^{o(\ell)}$.*
- *If the non-uniform ETH is true then META is* NP*-hard and META $\notin \bigcap_{\varepsilon>0} \mathrm{DTime}(2^{\varepsilon\cdot\ell})$.*

The lower bounds remain true even if Ψ is a union of self-join-free and acyclic conjunctive queries over a binary signature (that is, of arity 2).

We make some remarks about Theorem 5. First, it may seem counterintuitive that the algorithmic part of this result relies on some lower bound conjectures. This is explained by the fact that an algorithmic result for META is actually a classification result for the underlying counting problem. The lower bound conjectures are the

[1]We remark that [11, Theorem 7] focuses on the special case of graphs and *near* linear time algorithms. However, in the word RAM model with $O(\log n)$ bits, a linear time algorithm is possible [20].
[2]A conjunctive query is semantically acyclic if and only if its #core (Definition 16) is acyclic.

reason that the algorithm for Meta can answer that a linear-time algorithm is *not* possible for certain UCQs.

Second, while for counting the answers to a CQ in linear time the property of being acyclic is the right criterion, note that for unions of CQs, acyclicity is not even sufficient for tractability. Even when restricted to unions of acyclic conjunctive queries, the meta problem is NP-hard.

Third, we elaborate on the idea that we use to prove Theorem 5. As mentioned before, the algorithmic part of Theorem 5 comes from the well-known technique of expressing UCQ answer counts in terms of linear combinations of CQ answer counts, and establishing a corresponding complexity monotonicity property, see Section 2.3. The more interesting result is the hardness part. Here we discover a connection between the meta question stated in Meta, and a topological invariant, namely, the question whether the reduced Euler characteristic of a simplicial complex is non-zero. It is known that simplicial complexes with non-vanishing reduced Euler characteristic are evasive, and as such this property is also related to Karp's Evasiveness Conjecture (see e.g. the excellent survey of Miller [53]). We use the known fact that deciding whether the reduced Euler characteristic is vanishing is NP-hard [58]. Roughly, the reduction works as follows. Given some simplicial complex Δ, we carefully define a UCQ Ψ_Δ in such a way that only one particular term in the CQ expansion of Ψ_Δ determines the linear-time tractability of counting answers to Ψ_Δ. However, the coefficient of this term is zero precisely if the reduced Euler characteristic of Δ is vanishing.

Simplicial complexes also appeared in a related context in a work by Roth and Schmitt [57]. They show a connection between the complexity of counting induced subgraphs that fulfil some graph property and the question whether a simplicial complex associated with this graph property is non-zero. To solve their problem, it suffices to consider simplicial *graph* complexes, which are special simplicial complexes whose elements are subsets of the edges of a complete graph, and to encode these as induced subgraph counting problems In contrast, to get our result we must encode arbitrary abstract simplicial complexes as UCQs and to show how to transfer the question about their Euler characteristic to a question about linear-time solvability of UCQs.

It turns out that, as additional consequences of our reduction in the proof of Theorem 5, we also obtain lower bounds for (approximately) computing the so-called Weisfeiler-Leman-dimension of a UCQ.

Consequences for the Weisfeiler-Leman-dimension of quantifier-free UCQs. During the last decade we have witnessed a resurge in the study of the *Weisfeiler-Leman-dimension* of graph classes and graph parameters [4, 9, 30, 36, 46, 54]. The Weisfeiler-Leman algorithm (WL-algorithm) and its higher-dimensional generalisations are important heuristics for graph isomorphism; for example, the 1-dimensional WL-algorithm is equivalent to the method of colour-refinement. We refer the reader to e.g. the EATCS Bulletin article of Arvind [4] for a concise and self-contained introduction; however, in this work we will use the WL-algorithm only in a black-box manner.

For each positive integer k, we say that two graphs G_1 and G_2 are *k-WL equivalent*, denoted by $G_1 \cong_k G_2$, if they cannot be distinguished by the k-dimensional WL-algorithm. A graph parameter π is called *k-WL invariant* if $G_1 \cong_k G_2$ implies $\pi(G_1) = \pi(G_2)$. Moreover, the *WL-dimension* of π is the minimum k for which π is k-WL invariant, if such a k exists, and ∞ otherwise (see e.g.[5]). The WL-dimension of a graph parameter π provides important information about the descriptive complexity of π [18]. Moreover, recent work of Morris et al. [54] shows that the WL-dimension of a graph parameter lower bounds the minimum dimension of a higher-order Graph Neural Network that computes the parameter.

The definitions of the WL-algorithm and the WL-dimension extend from graphs to labelled graphs, that is, directed multi-graphs with edge- and vertex-labels (see e.g. [49]). Formally, we say that a database is a *labelled graph* if its signature has arity at most 2, and if it contains no self-loops, that is, tuples of the form (v, v). Similarly, *(U)CQs on labelled graphs* have signatures of arity at most 2 and contain no atom of the form $R(v, v)$.

Definition 6 (WL-dimension)**.** Let Ψ be a UCQ on labelled graphs. The *WL-dimension* of Ψ, denoted by $\dim_{\mathrm{WL}}(\Psi)$, is the minimum k such that, for any pair of labelled graphs $\mathcal{D}_1$ and $\mathcal{D}_2$ with $\mathcal{D}_1 \cong_k \mathcal{D}_2$, it holds that the number of answers to Ψ in $\mathcal{D}_1$ is the same as in $\mathcal{D}_2$. If no such k exists, then the WL-dimension is ∞.

Note that a CQ is a special case of a UCQ, so Definition 6 also applies when Ψ is a CQ φ.

It was shown very recently that the WL-dimension of a *quantifier-free conjunctive query* φ on labelled graphs is equal to the treewidth of the Gaifman graph of φ [49, 55]. Using known algorithms for computing the treewidth [16, 33] it follows that, for every fixed positive integer d, the problem of deciding whether the WL-dimension of φ is at most d can be solved in polynomial time (in the size of φ). Moreover, the WL-dimension of φ can be efficiently approximated in polynomial time. In stark contrast, we show that the computation of the WL-dimension of a *UCQ* is much harder; in what follows, we say that S is an *f-approximation* of k if $k \le S \le f(k) \cdot k$.

Theorem 7. *There is an algorithm that computes a $O(\sqrt{\log k})$-approximation of the WL-dimension k of a quantifier-free UCQ on labelled graphs $\Psi = \varphi_1 \vee \cdots \vee \varphi_\ell$ in time $|\Psi|^{O(1)} \cdot O(2^\ell)$.*

Moreover, let $f : \mathbb{Z}_{>0} \to \mathbb{Z}_{>0}$ be any computable function. The problem of computing an f-approximation of $\dim_{\mathrm{WL}}(\Psi)$ given an input UCQ $\Psi = \varphi_1 \vee \cdots \vee \varphi_\ell$ is NP-hard, and, assuming ETH, an f-approximation of $\dim_{\mathrm{WL}}(\Psi)$ cannot be computed in time $2^{o(\ell)}$.

Finally, the computation of the WL-dimension of UCQs stays intractable even if we fix k.

Theorem 8. *Let k be any* fixed *positive integer. The problem of deciding whether the WL-dimension of a quantifier-free UCQ on labelled graphs $\Psi = \varphi_1 \vee \cdots \vee \varphi_\ell$ is at most k can be solved in time $|\Psi|^{O(1)} \cdot O(2^\ell)$.*

Moreover, the problem is NP*-hard and, assuming ETH, cannot be solved in time $2^{o(\ell)}$.*

1.2 Further Related Work

For exact counting it makes a substantial difference whether one wants to count answers to a conjunctive query or a union of conjunctive queries [23, 31]. However, for approximate counting, unions can generally be handled using a standard trick of Karp and Luby [45], and therefore fixed-parameter tractability results for approximately counting the answers to a conjunctive query also extend to unions of conjunctive queries [3, 35].

Counting and enumerating the answers to a union of conjunctive queries has also been studied in the context of dynamic databases [13, 14]. This line of research investigates the question whether linear-time dynamic algorithms are possible. Concretely, the question is whether, after a preprocessing step that builds a data structure in time linear in the size of the initial database, the number of answers to a fixed union of conjunctive queries can be returned in constant time with a constant-time update to the data structure, whenever there is a change to the database. Berkholz et al. show that for a conjunctive query such a linear-time algorithm is possible if and only if the CQ is *q-hierarchical* [13, Theorem 1.3]. There are acyclic CQs that are not q-hierarchical, for instance the query $\varphi(\{a,b,c,d\}) = E(a,b) \land E(b,c) \land E(c,d)$ is clearly acyclic — however, the sets of atoms that contain b and c, respectively, are neither comparable nor disjoint, and therefore φ is not q-hierarchical. So, there are queries for which counting in the static setting is easy, whereas it is hard in the dynamic setting. Berkholz et al. extend their result from CQs to UCQs [14, Theorem 4.5], where the criterion is whether the UCQ is *exhaustively q-hierarchical*. This property essentially means that, for every subset of the CQs in the union, if instead of taking the disjunction of these CQs we take the conjunction, then the resulting CQ should be q-hierarchical. Moreover, checking whether a CQ ϕ is q-hierarchical can be done in time polynomial in the size of ϕ. However, the straightforward approach of checking whether a UCQ is exhaustively q-hierarchical takes exponential time, and it is stated as an open problem in [14] whether this can be improved. In the dynamic setting this question remains open — however, in the static setting we show that, while for counting answers to CQs the criterion for linear-time tractability can be verified in polynomial time, this is not true for unions of conjunctive queries, subject to some complexity assumptions, as we have seen in Theorem 5.

1.3 Organisation of the Paper

The subsequent Section 2 introduces some preliminary material, and in Section 3 we prove the complexity classification of #UCQ(C) for deletion-closed classes C. Due to the space constraints, we defer the treatment of Meta and its connection to the WL-dimension to the appendix.

2 PRELIMINARIES

Due to the fine-grained nature of the questions we ask in this work (e.g. linear time counting vs non-linear time counting), it is important to specify the machine model. We use the standard word RAM model with $O(\log n)$ bits. The exact model makes a difference. For example, it is possible to count answers to quantifier-free acyclic conjunctive queries in linear time in the word RAM model [20], while Turing machines only achieve near linear time (or expected linear time) [11].

Due to the space constraints, we defer the introduction of some background material on parameterised complexity theory and relational databases to the appendix.

2.1 Fine-grained Complexity Theory

In this work, we will rely on the following hypotheses from fine-grained complexity theory.

Conjecture 9 (ETH [43]). *3-SAT cannot be solved in time* $\exp(o(n))$, *where* n *denotes the number of variables of the input formula.*

Conjecture 10 (SETH [19, 43]). *For each* $\varepsilon > 0$ *there exists a positive integer* k *such that* k*-SAT cannot be solved in time* $O(2^{(1-\varepsilon)n})$, *where* n *denotes the number of variables of the input formula.*

Conjecture 11 (Non-uniform ETH [26]). *3-SAT is not contained in* $\bigcap_{\varepsilon>0} \mathrm{DTime}(\exp(\varepsilon n))$, *where* n *denotes the number of variables of the input formula.*

Conjecture 12 (Triangle Conjecture [1]). *There exists* $\gamma > 0$ *such that any (randomised) algorithm that decides whether a graph with* n *vertices and* m *edges contains a triangle takes time at least* $\Omega(m^{1+\gamma})$ *in expectation.*

2.2 Homomorphisms and Conjunctive Queries

We assume familiarity with the central notions of relational databases such as signatures, structures, and Gaifman graphs. We refer the reader to Appendix A.2 for a brief introduction.

Homomorphisms as Answers to CQs. Let $\mathcal{A}$ and $\mathcal{B}$ be structures over signatures $\tau_{\mathcal{A}} \subseteq \tau_{\mathcal{B}}$. A *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$ is a mapping $h : U(\mathcal{A}) \to U(\mathcal{B})$ such that for each relation symbol $R \in \tau_{\mathcal{A}}$ with arity a and each tuple $\vec{t} = (t_1, \dots, t_a) \in R^{\mathcal{A}}$ we have that $h(\vec{t}) = (h(t_1), \dots, h(t_a)) \in R^{\mathcal{B}}$. We use $\mathsf{Hom}(\mathcal{A} \to \mathcal{B})$ to denote the set of homomorphisms from $\mathcal{A}$ to $\mathcal{B}$, and we use the lower case version $\mathsf{hom}(\mathcal{A} \to \mathcal{B})$ to deonote the *number* of homomorphisms from $\mathcal{A}$ to $\mathcal{B}$.

Let φ be a conjunctive query with free variables $X = \{x_1, \dots, x_k\}$ and quantified variables $Y = \{y_1, \dots, y_d\}$. We can associate φ with a structure $\mathcal{A}_\varphi$ defined as follows: The universe of $\mathcal{A}_\varphi$ are the variables $X \cup Y$ and for each atom $R(\vec{t})$ of φ we add the tuple $\vec{t}$ to $R^{\mathcal{A}}$. It is well-known that, for each database $\mathcal{D}$, the set of answers of φ in $\mathcal{D}$ is precisely the set of assignments $a : X \to U(\mathcal{D})$ such that there is a homomorphism $h \in \mathsf{Hom}(\mathcal{A}_\varphi \to \mathcal{D})$ with $h|_X = a$. Since working with (partial) homomorphisms will be very convenient in this work, we will use the notation from [31] and (re)define a conjunctive query as a pair consisting of a relational structure $\mathcal{A}$ together with a set $X \subseteq U(\mathcal{A})$. The size of $(\mathcal{A}, X)$ is denoted by $|(\mathcal{A}, X)|$ and defined to be $|\mathcal{A}| + |X|$. Furthermore, the set of answers of $(\mathcal{A}, X)$ in $\mathcal{D}$, denoted by $\mathsf{Ans}((\mathcal{A}, X) \to \mathcal{D})$, is defined as $\{a : X \to U(\mathcal{D}) \mid \exists h \in \mathsf{Hom}(\mathcal{A} \to \mathcal{D}) : h|_X = a\}$. We then use $\mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$ to denote the number of answers, i.e., $\mathsf{ans}((\mathcal{A}, X) \to \mathcal{D}) := |\mathsf{Ans}((\mathcal{A}, X) \to \mathcal{D})|$.

We can now formally define the (parameterised) problem of counting answers to conjunctive queries. As is usual, we restrict the problem by a class C of allowed queries.

Name: #CQ(C)
Input: A conjunctive query $(\mathcal{A}, X) \in C$ and a database $\mathcal{D}$.
Parameter: $|(\mathcal{A}, X)|$.
Output: The number of answers $\mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$.

#Equivalence and #Minimality. In the realm of decision problems, it is well known that evaluating a conjunctive query is equivalent

to evaluating the (homomorphic) core of the query, i.e., evaluating the minimal homomorphic-equivalent query. A similar, albeit slightly different notion of equivalence and minimality is required for counting answers to conjunctive queries. In what follows, we will provide the necessary definitions and properties of equivalence, minimality and cores for counting answers to conjunctive queries, and we refer the reader to [23] and to the full version of [31] for a more comprehensive discussion. To avoid confusion between the notions in the realms of decision and counting, we will from now on use the # symbol for the counting versions (see Definition 14).

Definition 13. Two conjunctive queries $(\mathcal{A}, X)$ and $(\mathcal{A}', X')$ are *isomorphic*, denoted by $(\mathcal{A}, X) \cong (\mathcal{A}', X')$, if there is an isomorphism b from $\mathcal{A}$ to $\mathcal{A}'$ with $b(X) = X'$.

Definition 14 (#Equivalence and #minimality (see [23, 31])). Two conjunctive queries $(\mathcal{A}, X)$ and $(\mathcal{A}', X')$ are *#equivalent*, denoted by $(\mathcal{A}, X) \sim (\mathcal{A}', X')$, if for every database $\mathcal{D}$ we have $\mathsf{ans}((\mathcal{A}, X) \to \mathcal{D}) = \mathsf{ans}((\mathcal{A}', X') \to \mathcal{D})$. A conjunctive query $(\mathcal{A}, X)$ is *#minimal* if there is no proper substructure $\mathcal{A}'$ of $\mathcal{A}$ such that $(\mathcal{A}, X) \sim (\mathcal{A}', X)$.

Observation 15. *The following are equivalent:*

(1) A conjunctive query $(\mathcal{A}, X)$ is #minimal.
(2) $(\mathcal{A}, X)$ has no #equivalent substructure that is induced by a set U with $X \subseteq U \subset U(\mathcal{A})$.
(3) Every homomorphism from $\mathcal{A}$ to itself that is the identity on X is surjective.

It turns out that #equivalence is the same as isomorphism if all variables are free, and it is the same as homomorphic equivalence if all variables are existentially quantified (see e.g. the discussion in Section 5 in the full version of [31]). Moreover, each quantifier-free conjunctive query is #minimal.

Definition 16 (#core). A *#core* of a conjunctive query $(\mathcal{A}, X)$ is a #minimal conjunctive query $(\mathcal{A}', X')$ with $(\mathcal{A}, X) \sim (\mathcal{A}', X')$.

It is well known (see e.g. [31]) that,for #minimal queries, #equivalence and isomorphism coincide. Thus the #core is unique up to isomorphisms; in fact, this allows us to speak of "the" #core of a conjunctive query.

Classification of #CQ(C) via Treewidth and Contracts. It is well known that the complexity of counting answers to a conjunctive query is governed by its treewidth, and by the treewidth of its contract [22, 31], which we define as follows.

Definition 17 (Contract). Let $(\mathcal{A}, X)$ be a conjunctive query, let $Y = U(\mathcal{A}) \setminus X$, and let G be the Gaifmann graph of $\mathcal{A}$. The *contract* of $(\mathcal{A}, X)$, denoted by $\mathsf{contract}(\mathcal{A}, X)$ is obtained from $G[X]$ by adding an edge between each pair of vertices u and v for which there is a connected component S in $G[Y]$ that is adjacent to both u and v, that is, there are vertices $x, y \in S$ such that $\{x, u\} \in E(G)$ and $\{y, v\} \in E(G)$. Given a class of conjunctive queries C, we write $\mathsf{contract}(C)$ for the class of all contracts of queries in C.

We note that there are multiple equivalent ways to define the contract of a query. For our purposes, the definition in [31] is most suitable. Also, the treewidth of the contract of a conjunctive query is an upper bound of what is called the query's "star size" in [32] and its "dominating star size" in [31].

Chen and Mengel established the following classification for counting answers to conjunctive queries of bounded arity.

Theorem 18 ([22]). *Let C be a recursively enumerable class of conjunctive queries of bounded arity, and let C' be the class of #cores of queries in C. If the treewidth of C' and of* $\mathsf{contract}(C')$ *is bounded, then #CQ(C) is solvable in polynomial time. Otherwise, #CQ(C) is* W[1]*-hard.*

We point out that the W[1]-hard cases can further be partitioned into W[1]-complete, #W[1]-complete and even harder cases [22, 31].[3] However, for the purpose of this work, we are only interested in tractable and intractable cases (recall that W[1]-hard problems are not fixed-parameter tractable under standard assumptions from fine-grained and parameterised complexity theory, such as ETH).

Self-join-free Conjunctive Queries and Isolated Variables. A conjunctive query $(\mathcal{A}, X)$ is *self-join-free* if each relation of $\mathcal{A}$ contains at most one tuple. We say that a variable of a conjunctive query is *isolated* if it is not part of any relation.

Note that that adding/removing isolated variables to/from a conjunctive query does not change its treewidth or the treewidth of its #core. Further, it does not change the complexity of counting answers: Just multiply/divide by n^v, where n is the number of elements of the database and v it the number of added/removed isolated free variables. For this reason, we will allow ourselves in this work to freely add and remove isolated variables from the queries that we encounter. For the existence of homomorphisms we also observe the following.

Observation 19. *Let $(\mathcal{A}, X)$ be a conjunctive query, let X' be a superset of X and let $\mathcal{A}'$ be the structure obtained from $\mathcal{A}$ by adding an isolated variable for each $x \in X' \setminus X$. Then for all $a : X' \to U(\mathcal{D})$ we have that $a|_X \in \mathsf{Ans}((\mathcal{A}, X) \to \mathcal{D})$ iff $a \in \mathsf{Ans}((\mathcal{A}', X') \to \mathcal{D})$.*

2.3 UCQs and the Homomorphism Basis

A *union of conjunctive queries* (UCQ) Ψ is a tuple of structures $(\mathcal{A}_1, \dots, \mathcal{A}_{\ell(\Psi)})$ over the same signature together with a set of designated elements X (the free variables) that are in the universe of each of the structures. For each $J \subseteq [\ell(\Psi)]$, we define $\Psi|_J = ((A_j)_{j \in J}, X)$. If we restrict to a single term of the union then we usually just write Ψ_i instead of $\Psi|_{\{i\}}$. Note that $\Psi_i = (\mathcal{A}_i, X)$ is simply a conjunctive query (rather than a union of CQs).

We will assume (without loss of generality) that, for any distinct i and i' in $[\ell(\Psi)]$, $U(\mathcal{A}_i) \cap U(\mathcal{A}_{i'}) = X$, i.e., that each CQ in the union has its own set of existentially quantified variables.

If each such conjunctive query is acyclic we say that Ψ is a union of acyclic conjunctive queries. Moreover, the arity of Ψ is the maximum arity of any of the $\mathcal{A}_i$. The size of Ψ is $|\Psi| = \sum_{i=1}^{\ell(\Psi)} |\Psi_i|$. The elements of X are the *free variables* of Ψ and $\ell(\Psi)$ is the number of CQs in the union. The set of *answers* of Ψ in a database $\mathcal{D}$, denoted by $\mathsf{Ans}(\Psi \to \mathcal{D})$ is defined as follows:

$$\mathsf{Ans}(\Psi \to \mathcal{D}) = \{a : X \to U(\mathcal{D}) \mid \exists i \in [\ell] : a \in \mathsf{Ans}(\Psi_i \to \mathcal{D})\}\,.$$

Again, we use the lower case version $\mathsf{ans}(\Psi \to \mathcal{D})$ to denote the *number* of answers of Ψ in $\mathcal{D}$.

[3] Those cases are: #W[2]-hard and #A[2]-complete.

In the definition of UCQs we assume that every CQ in the union has the same set of free variables, namely X. This assumption is without loss of generality. To see this, suppose that we have a union of CQs $(\mathcal{A}_1, X_1), \dots, (\mathcal{A}_\ell, X_\ell)$ with individual sets of free variables. Let $X = \bigcup_{i=1}^{\ell} X_i$ and, for each $i \in [\ell]$, let $\mathcal{A}'_i$ be the structure obtained from $\mathcal{A}_i$ by adding an isolated variable for each $x \in X \setminus X_i$. Then consider the UCQ $\Psi := ((\mathcal{A}'_1, \dots, \mathcal{A}'_\ell), X)$. If for some assignment $a : X \to U(\mathcal{D})$ it holds that there is an $i \in [\ell]$ such that $a|_{X_i} \in \mathsf{Ans}((\mathcal{A}_i, X_i) \to \mathcal{D})$. Then, according to Observation 19, this is equivalent to $a \in \mathsf{Ans}((\mathcal{A}'_i, X) \to \mathcal{D})$, which means that a is an answer of Ψ. So, without loss of generality we can work with Ψ, which uses the same set of free variables for each CQ in the union.

Now we define the parameterised problem of counting answers to UCQs. As usual, the problem is restricted by a class C of allowed queries with respect to which we classify the complexity.

Name: #UCQ(C)
Input: A UCQ $\Psi \in C$ together with a database $\mathcal{D}$.
Parameter: $|\Psi|$.
Output: The number of answers $\mathsf{ans}(\Psi \to \mathcal{D})$.

The next definition will be crucial for the analysis of the complexity of #UCQ(C).

Definition 20 (combined query $\wedge(\Psi)$)**.** Let $\Psi = ((\mathcal{A}_1, , \dots, \mathcal{A}_\ell), X)$ be a UCQ. Then we define the *combined query* $\wedge(\Psi) = (\bigcup_{j\in[\ell]} A_j, X)$.

What follows is an easy, but crucial observation about $\wedge(\Psi|_J)$.

Observation 21. *Let $((\mathcal{A}_1, , \dots, \mathcal{A}_\ell), X)$ be a UCQ, and let $\emptyset \neq J \subseteq [\ell]$. For each database $\mathcal{D}$ and assignment $a : X \to U(\mathcal{D})$ we have*

$$a \in \mathsf{Ans}(\wedge(\Psi|_J) \to \mathcal{D}) \Leftrightarrow \forall j \in J : a \in \mathsf{Ans}(\Psi_j \to \mathcal{D}).$$

Definition 22 (Coefficient function c_Ψ)**.** Let $\Psi = ((\mathcal{A}_1, , \dots, \mathcal{A}_\ell), X)$ be a UCQ. For each conjunctive query $(\mathcal{A}, X)$, we set $\mathcal{I}(\mathcal{A}, X) = \{J \subseteq [\ell] \mid (\mathcal{A}, X) \sim \wedge(\Psi|_J)\}$, and we define the *coefficient function* of Ψ as follows: $c_\Psi(\mathcal{A}, X) = \sum_{J\in\mathcal{I}(\mathcal{A},X)} (-1)^{|J|+1}$.

Using inclusion-exclusion, we can transform the problem of counting answers to Ψ into the problem of evaluating a linear combination of CQ answer counts. We include a proof in Appendix A.3 only for reasons of self-containment and note that the complexity-theoretic applications of this transformation, especially regarding lower bounds, have first been discovered by Chen and Mengel [23].

Lemma 23 ([23])**.** *Let $\Psi = ((\mathcal{A}_1, , \dots, \mathcal{A}_\ell), X)$ be a UCQ. For every database $\mathcal{D}$, $\mathsf{ans}(\Psi \to \mathcal{D}) = \sum_{(\mathcal{A},X)} c_\Psi(\mathcal{A}, X) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$, where the sum is over all equivalence classes of $\sim$.*

We conclude this subsection with the following two operations on classes of UCQs.

Definition 24 ($\Gamma(C)$ and $\wedge(C)$)**.** Let C be a class of UCQs. $\Gamma(C)$ is the class of all $(\mathcal{A}, X)$ such that $(\mathcal{A}, X)$ is #minimal and there is $\Psi \in C$ with $c_\Psi(\mathcal{A}, X) \neq 0$. Let $\wedge(C) = \{\wedge(\Psi) \mid \Psi \in C\}$.

It was independently discovered by Chen and Mengel [23], and by Curticapean, Dell and Marx [27] that the computation of a linear combination of homomorphism counts is precisely as hard as computing its hardest term. Moreover, in the former work, Chen and Mengel also established this property in the more general context of linear combinations of conjunctive queries. Applying this principle to counting answers to UCQs (which we have seen to be equivalent to computing a linear combination in Lemma 23), we obtain the following two results; details are provided in Appendix A.4.

Corollary 25. *Let Ψ be a UCQ. For each $d \geq 1$, computing the function $\mathcal{D} \mapsto \mathsf{ans}(\Psi \to \mathcal{D})$ can be done in time $O(|\mathcal{D}|^d)$ if and only if for each #minimal $(\mathcal{A}, X)$ with $c_\Psi(\mathcal{A}, X) \neq 0$ the function $\mathcal{D} \mapsto \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$ can be computed in time $O(|\mathcal{D}|^d)$.*

Corollary 26 (Implicitly also in [23])**.** *Let C be a recursively enumerable class of UCQs. The problems #UCQ(C) and #CQ($\Gamma(C)$) are interreducible with respect to parameterised Turing-reductions.*

3 PROOFS OF THEOREM 2 AND THEOREM 3

Let C be a class of UCQs. Recall that $\wedge(C)$ is the class of all conjunctive queries that are obtained just by substituting all $\vee$ by $\wedge$ in UCQs in C, whereas $\Gamma(C)$ in Theorem 1 is the much less natural class of #minimal queries that survive with a non-zero coefficient in the CQ expansion of a UCQ in C. The work of Chen and Mengel [23] implicitly also shows an upper bound for counting answers to UCQs from the class C in terms of the simpler objects $\wedge(C)$ and $\mathsf{contract}(\wedge(C))$, rather than in terms of the more complicated objects $\Gamma(C)$ and $\mathsf{contract}(\Gamma(C))$. We include a proof for completeness.

Lemma 27. *Let C be recursively enumerable class of UCQs. Suppose that both $\wedge(C)$ and $\mathsf{contract}(\wedge(C))$ have bounded treewidth. Then #UCQ(C) is fixed-parameter tractable.*

Proof. Let $\Psi \in C$. Recall from the proof of Lemma 23 that, for every $\mathcal{D}$, $\mathsf{ans}(\Psi \to \mathcal{D}) = \sum_{\emptyset\neq J\subseteq[\ell]} (-1)^{|J|+1} \cdot \mathsf{hom}(\wedge(\Psi|_J) \to \mathcal{D})$. Hence #UCQ($C$) $\leq^{\mathsf{FPT}}$ #CQ($\hat{C}$) where $\hat{C}$ is $\{\wedge(\Psi|_J) \mid \Psi \in C \wedge \emptyset \neq J \subseteq [\ell(\Psi)]\}$. Finally, since $\wedge(\Psi|_J)$ is a subquery of $\wedge(\Psi)$ for each J, the treewidths of $\wedge(\Psi|_J)$ and $\mathsf{contract}(\wedge(\Psi|_J))$ are bounded from above by the treewidths of $\wedge(\Psi)$ and $\mathsf{contract}(\wedge(\Psi))$, respectively. Consequently, the treewidths of $\hat{C}$ and $\mathsf{contract}(\hat{C})$ are bounded, and thus #CQ($\hat{C}$) is polynomial-time solvable by the classification of Chen and Mengel [22, Theorem 22] Since #UCQ(C) $\leq^{\mathsf{FPT}}$ #CQ($\hat{C}$), the lemma follows. □

Our goal is to relate the complexity of #UCQ(C) to the structure of $\wedge(C)$ with the hope of obtaining a more natural tractability criterion than the one given by Theorem 1. While we will see later that this seems not always possible (Appendix E), we identify conditions under which a natural criterion based on $\wedge(C)$ is possible, both in the quantifier-free case (Section 3.1), and in the general case that allows quantified variables (Section 3.2).

A class of UCQs C is *closed under deletions* if, for every $\Psi = ((\mathcal{A}_1, \dots, \mathcal{A}_\ell), X) \in C$ and for every $\emptyset \neq J \subseteq [\ell]$, the UCQ $\Psi|_J$ is also contained in C. For example, any class of UCQs defined by prescribing the allowed conjunctive queries is closed under deletions. This includes, e.g., unions of acyclic conjunctive queries.

3.1 The Quantifier-free Case

As a warm-up, we start with the much simpler case of quantifier-free queries. Here, we only allow (unions of) conjunctive queries $(\mathcal{A}, X)$ satisfying $U(\mathcal{A}) = X$.

Lemma 28. *Let C be a recursively enumerable class of quantifier-free UCQs of bounded arity. Suppose that C is closed under deletions. If $\wedge(C)$ has unbounded treewidth then #UCQ(C) is* W[1]*-hard.*

Proof. We show that $\wedge(C) \subseteq \Gamma(C)$, which then proves the claim by Theorem 1. Recall from Definition 24 that $\Gamma(C) = \{(\mathcal{A}, X) \mid (\mathcal{A}, X)$ is #minimal and there is $\Psi \in C$ with $c_\Psi(\mathcal{A}, X) \neq 0\}$. Let $\Psi = ((\mathcal{A}_1, \ldots, \mathcal{A}_\ell), X) \in C$. Note that, according to Observation 15, $\wedge(\Psi)$ is its own #core since it does not have existentially quantified variables. For the same reason, for each nonempty subset J of $[\ell]$, the query $\wedge(\Psi|_J)$ is its own #core. Now let $J \subseteq [\ell]$ be inclusion-minimal with the property that $\wedge(\Psi|_J)$ is isomorphic to $\wedge(\Psi)$. Since C is closed under deletions, the UCQ $\Psi|_J$ is contained in C. By the inclusion-minimality of J, Definition 22 ensures that $c_{\Psi|_J}(\wedge(\Psi)) = (-1)^{|J|+1} \neq 0$. As a consequence, $\wedge(\Psi) \in \Gamma(C)$, concluding the proof. □

From Lemmas 28 and 27 together with the fact that the contract of a quantifier-free query is the query itself, we obtain Theorem 2.

3.2 The General Case

Now we consider UCQs with existentially quantified variables. Here, a corresponding hardness result (Lemma 31) can be achieved under some additional assumptions. Note that the number of existentially quantified variables in a UCQ $\Psi = ((\mathcal{A}_1, \ldots, \mathcal{A}_\ell), X)$ is equal to $\sum_{i=1}^{\ell} |U(\mathcal{A}_i) \setminus X|$. We first need the following two auxiliary results:

Lemma 29. *Let $(\mathcal{A}, X)$ and $(\mathcal{A}', X')$ be #equivalent conjunctive queries. Further, let G and G' be the Gaifman graphs of $\mathcal{A}$ and $\mathcal{A}'$, respectively. Then $G[X]$ and $G'[X']$ are isomorphic.*

Lemma 30. *Let $(\mathcal{A}, X)$ be a self-join-free conjunctive query. Let $\mathcal{A}'$ be the structure obtained from $\mathcal{A}$ by deleting all isolated variables in $U(\mathcal{A}) \setminus X$. Then $(\mathcal{A}', X)$ is the #core of $(\mathcal{A}, X)$.*

Lemma 31. *Let C be a recursively enumerable class of unions of self-join-free conjunctive queries with bounded arity. Suppose that C is closed under deletions and that there is a finite upper bound on the number of existentially quantified variables in queries in C. If either of $\wedge(C)$ or* contract$(\wedge(C))$ *have unbounded treewidth then #UCQ(C) is* W[1]*-hard.*

Proof. Let d be the maximum number of existentially quantified variables in a query in C. Assume first that $\wedge(C)$ has unbounded treewidth. We show that $\Gamma(C)$ has unbounded treewidth, which proves the claim by Theorem 1. To this end, let B be any positive integer. The goal is to find a conjunctive query in $\Gamma(C)$ with treewidth at least B. Since $\wedge(C)$ has unbounded treewidth, there is a UCQ $\Psi = ((\mathcal{A}_1, \ldots, \mathcal{A}_\ell), X)$ in C such that $\wedge(\Psi)$ has treewidth larger than $d + B$. Note that, although all Ψ_i are self-join-free, $\wedge(\Psi)$ is not necessarily self-join-free. Let J be inclusion-minimal among the subsets of $[\ell]$ with the property that the #core of $\wedge(\Psi|_J)$ is isomorphic to the #core of $\wedge(\Psi)$. Since C is closed under deletions, the UCQ $\Psi|_J$ is contained in C. Let $(\mathcal{A}', X')$ be the #core of $\wedge(\Psi)$.

By inclusion-minimality of J, $c_{\Psi|_J}((\mathcal{A}', X')) = (-1)^{|J|+1} \neq 0$. As a consequence, $(\mathcal{A}', X') \in \Gamma(C)$. It remains to show that the treewidth of $(\mathcal{A}', X')$ is at least B. For this, let G be the Gaifmann graph of $\wedge(\Psi)$ and let G' be the Gaifmann graph of the #core of $\wedge(\Psi)$ (the Gaifman graph of $\mathcal{A}'$). First, deletion of a vertex can decrease the treewidth by at most 1. Thus, $G[X]$ has treewidth at least $d + B - d = B$. By Lemma 29, $G[X]$ and $G'[X']$ are isomorphic.

Therefore the treewidth of $G'[X']$, i.e., the treewidth of $(\mathcal{A}', X')$, is at least B. So we have shown that if the treewidth of $\wedge(C)$ is unbounded then so is the treewidth of $\Gamma(C)$.

In the second case, we assume that the contracts of queries in $\wedge(C)$ (see Definition 17) have unbounded treewidth. We introduce the following terminology: Let $(\mathcal{A}, X)$ be a conjunctive query and let $y \in U(\mathcal{A})$. The *degree of freedom* of y is the number of vertices in X that are adjacent to y in the Gaifman graph of $\mathcal{A}$. Let $\hat{C}$ be the class of all conjunctive queries $(\mathcal{A}, X)$ such that there exists $\Psi = ((\mathcal{A}_1, \ldots, \mathcal{A}_{\ell(\Psi)}), X)$ in C with $(\mathcal{A}, X) = (\mathcal{A}_i, X)$ for some $i \in [\ell(\Psi)]$. Since C is closed under deletions, $\hat{C} \subseteq C$. By the assumptions of the lemma, $\hat{C}$ consists only of self-join-free queries. Thus, by Lemma 30, each query in $\hat{C}$ is its own #core (up to deleting isolated variables). We will now consider the following cases:

(i) Suppose that $\hat{C}$ has unbounded degree of freedom. With Definition 17 it is straightforward to check that a quantified variable y with degree of freedom B induces a clique of size B in the contract of the corresponding query. Therefore, the contracts of the queries in $\hat{C}$ have unbounded treewidth. Consequently, #CQ($\hat{C}$) is W[1]-hard by the classification of Chen and Mengel [22, Theorem 22]. Since $\hat{C} \subseteq C$ the problem #CQ($\hat{C}$) is merely a restriction of #UCQ(C), the latter of which is thus W[1]-hard as well.

(ii) Suppose that the degree of freedom of queries in $\hat{C}$ is bounded by a constant d'. We show that $\Gamma(C)$ has unbounded treewidth, which proves the claim by Theorem 1. To this end, let B be any positive integer. The goal is to find a conjunctive query in $\Gamma(C)$ with treewidth at least B. Since contract$(\wedge(C))$ has unbounded treewidth, there is a UCQ $\Psi = (\mathcal{A}_1, \ldots, \mathcal{A}_\ell), X)$ in C such that contract$(\wedge(\Psi))$ has treewidth larger than $d + \binom{dd'}{2} + B$. We will show that $\wedge(\Psi)$ has treewidth larger than $d + B$, which, as we have argued previously, implies that $\Gamma(C)$ contains a query with treewidth at least B. To prove that $\wedge(\Psi)$ indeed has treewidth larger than $d + B$, let $\wedge(\Psi) = (\mathcal{A}, X)$ and let G be the Gaifman graph of $\mathcal{A}$. Let $Y = U(\mathcal{A}) \setminus X$ and recall from Definition 17 that contract$(\mathcal{A}, X)$ is obtained from $G[X]$ by adding an edge between any pair of vertices u and v that are adjacent to a common connected component in $G[Y]$. Let $N \subseteq X$ be the set of all vertices in X that are adjacent to a vertex in Y and observe that $|N| \leq dd'$ since the number of existentially quantified variables and the degree of freedom are bounded by d and d', respectively. Thus, contract$(\mathcal{A}, X)$ is obtained by adding at most $\binom{dd'}{2}$ edges to $G[X]$. The deletion of an edge can decrease the treewidth by at most 1, so $\mathrm{tw}(\wedge(\Psi)) = \mathrm{tw}(G) \geq \mathrm{tw}(G[X]) \geq \mathrm{tw}(\mathrm{contract}(\mathcal{A}, X)) - \binom{dd'}{2} > d + B$, which concludes Case (ii).

With all cases concluded, the proof is completed. □

From Lemmas 27 and 31 we directly obtain Theorem 3.

Remark 32. It turns out that all side conditions of Theorem 3 are necessary if we aim to classify #UCQ(C) solely via $\wedge(C)$. To this end, we provide counter examples for each missing condition in Appendix E.

REFERENCES

[1] Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 434–443. IEEE Computer Society, 2014.

[2] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[3] Marcelo Arenas, Luis Alberto Croquevielle, Rajesh Jayaram, and Cristian Riveros. When is approximate counting for conjunctive queries tractable? In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1015–1027. ACM, 2021.

[4] Vikraman Arvind. The Weisfeiler-Lehman procedure. *Bull. EATCS*, 120, 2016.

[5] Vikraman Arvind, Frank Fuhlbrück, Johannes Köbler, and Oleg Verbitsky. On the Weisfeiler-Leman dimension of fractional packing. *Inf. Comput.*, 288:104803, 2022.

[6] László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 684–697. ACM, 2016.

[7] Guillaume Bagan. *Algorithmes et complexité des problèmes d'énumération pour l'évaluation de requêtes logiques. (Algorithms and complexity of enumeration problems for the evaluation of logical queries).* PhD thesis, University of Caen Normandy, France, 2009.

[8] Guillaume Bagan, Arnaud Durand, and Etienne Grandjean. On acyclic conjunctive queries and constant delay enumeration. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 208–222. Springer, 2007.

[9] Pablo Barceló, Mikhail Galkin, Christopher Morris, and Miguel A. Romero Orth. Weisfeiler and Leman go relational. In Bastian Rieck and Razvan Pascanu, editors, *Learning on Graphs Conference, LoG 2022, 9-12 December 2022, Virtual Event*, volume 198 of *Proceedings of Machine Learning Research*, page 46. PMLR, 2022.

[10] Pablo Barceló, Miguel Romero, and Moshe Y. Vardi. Semantic Acyclicity on Graph Databases. *SIAM J. Comput.*, 45(4):1339–1376, 2016.

[11] Suman K. Bera, Lior Gishboliner, Yevgeny Levanzov, C. Seshadhri, and Asaf Shapira. Counting subgraphs in degenerate graphs. *J. ACM*, 69(3):23:1–23:21, 2022.

[12] Christoph Berkholz, Fabian Gerhardt, and Nicole Schweikardt. Constant delay enumeration for conjunctive queries: a tutorial. *ACM SIGLOG News*, 7(1):4–33, 2020.

[13] Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. Answering conjunctive queries under updates. In Emanuel Sallinger, Jan Van den Bussche, and Floris Geerts, editors, *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017*, pages 303–318. ACM, 2017.

[14] Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. Answering UCQs under updates and in the presence of integrity constraints. In Benny Kimelfeld and Yael Amsterdamer, editors, *21st International Conference on Database Theory, ICDT 2018, March 26-29, 2018, Vienna, Austria*, volume 98 of *LIPIcs*, pages 8:1–8:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[15] Christoph Berkholz and Nicole Schweikardt. Constant delay enumeration with fpt-preprocessing for conjunctive queries of bounded submodular width. In Peter Rossmanith, Pinar Heggernes, and Joost-Pieter Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 58:1–58:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[16] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.

[17] Johann Brault-Baron. *De la pertinence de l'énumération : complexité en logiques propositionnelle et du premier ordre. (The relevance of the list: propositional logic and complexity of the first order).* PhD thesis, University of Caen Normandy, France, 2013.

[18] Jin-yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992.

[19] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In Jianer Chen and Fedor V. Fomin, editors, *Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers*, volume 5917 of *Lecture Notes in Computer Science*, pages 75–85. Springer, 2009.

[20] Nofar Carmeli, Shai Zeevi, Christoph Berkholz, Alessio Conte, Benny Kimelfeld, and Nicole Schweikardt. Answering (unions of) conjunctive queries using random access and random-order enumeration. *ACM Trans. Database Syst.*, 47(3):9:1–9:49, 2022.

[21] Ashok K. Chandra and Philip M. Merlin. Optimal Implementation of Conjunctive Queries in Relational Data Bases. In John E. Hopcroft, Emily P. Friedman, and Michael A. Harrison, editors, *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 77–90. ACM, 1977.

[22] Hubie Chen and Stefan Mengel. A trichotomy in the complexity of counting answers to conjunctive queries. In Marcelo Arenas and Martín Ugarte, editors, *18th International Conference on Database Theory, ICDT 2015, March 23-27, 2015, Brussels, Belgium*, volume 31 of *LIPIcs*, pages 110–126. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[23] Hubie Chen and Stefan Mengel. Counting answers to existential positive queries: A complexity classification. In Tova Milo and Wang-Chiew Tan, editors, *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 315–326. ACM, 2016.

[24] Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David W. Juedes, Iyad A. Kanj, and Ge Xia. Tight lower bounds for certain parameterized NP-hard problems. *Inf. Comput.*, 201(2):216–231, 2005.

[25] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.*, 72(8):1346–1367, 2006.

[26] Yijia Chen, Kord Eickmeyer, and Jörg Flum. The exponential time hypothesis and the parameterized clique problem. In Dimitrios M. Thilikos and Gerhard J. Woeginger, editors, *Parameterized and Exact Computation - 7th International Symposium, IPEC 2012, Ljubljana, Slovenia, September 12-14, 2012. Proceedings*, volume 7535 of *Lecture Notes in Computer Science*, pages 13–24. Springer, 2012.

[27] Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 210–223. ACM, 2017.

[28] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

[29] Víctor Dalmau and Peter Jonsson. The complexity of counting homomorphisms seen from the other side. *Theor. Comput. Sci.*, 329(1-3):315–323, 2004.

[30] Holger Dell, Martin Grohe, and Gaurav Rattan. Lovász meets Weisfeiler and Leman. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 40:1–40:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[31] Holger Dell, Marc Roth, and Philip Wellnitz. Counting answers to existential questions. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 113:1–113:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[32] Arnaud Durand and Stefan Mengel. Structural tractability of counting of solutions to conjunctive queries. In Wang-Chiew Tan, Giovanna Guerrini, Barbara Catania, and Anastasios Gounaris, editors, *Joint 2013 EDBT/ICDT Conferences, ICDT '13 Proceedings, Genoa, Italy, March 18-22, 2013*, pages 81–92. ACM, 2013.

[33] Uriel Feige, Mohammad Taghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM J. Comput.*, 38(2):629–657, 2008.

[34] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.

[35] Jacob Focke, Leslie Ann Goldberg, Marc Roth, and Stanislav Zivný. Approximately counting answers to conjunctive queries with disequalities and negations. In Leonid Libkin and Pablo Barceló, editors, *PODS '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 315–324. ACM, 2022.

[36] Martin Fürer. On the combinatorial power of the weisfeiler-lehman algorithm. In Dimitris Fotakis, Aris Pagourtzis, and Vangelis Th. Paschos, editors, *Algorithms and Complexity - 10th International Conference, CIAC 2017, Athens, Greece, May 24-26, 2017, Proceedings*, volume 10236 of *Lecture Notes in Computer Science*, pages 260–271, 2017.

[37] Georg Gottlob, Gianluigi Greco, and Francesco Scarcello. Treewidth and hypertree width. In Lucas Bordeaux, Youssef Hamadi, and Pushmeet Kohli, editors, *Tractability: Practical Approaches to Hard Problems*, pages 3–38. Cambridge University Press, 2014.

[38] Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decomposition and tractable queries. *Journal of Computer and System Sciences*, 64(3):579–627, 2002.

[39] Gianluigi Greco and Francesco Scarcello. Counting solutions to conjunctive queries: structural and hybrid tractability. In Richard Hull and Martin Grohe, editors, *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'14, Snowbird, UT, USA, June 22-27, 2014*, pages 132–143. ACM, 2014.

[40] Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1:1–1:24, 2007.
[41] Martin Grohe and Dániel Marx. Constraint solving via fractional edge covers. *ACM Transactions on Algorithms*, 11(1):4:1–4:20, 2014.
[42] Martin Grohe, Thomas Schwentick, and Luc Segoufin. When is the evaluation of conjunctive queries tractable? In Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis, editors, *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 657–666. ACM, 2001.
[43] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.
[44] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.
[45] Richard M. Karp and Michael Luby. Monte-Carlo algorithms for enumeration and reliability problems. In *24th Annual Symposium on Foundations of Computer Science, Tucson, Arizona, USA, 7-9 November 1983*, pages 56–64. IEEE Computer Society, 1983.
[46] Sandra Kiefer, Ilia Ponomarenko, and Pascal Schweitzer. The Weisfeiler-Leman dimension of planar graphs is at most 3. *J. ACM*, 66(6):44:1–44:31, 2019.
[47] Phokion G. Kolaitis and Moshe Y. Vardi. On the expressive power of datalog: Tools and a case study. *J. Comput. Syst. Sci.*, 51(1):110–134, 1995.
[48] Phokion G. Kolaitis and Moshe Y. Vardi. Conjunctive-query containment and constraint satisfaction. *J. Comput. Syst. Sci.*, 61(2):302–332, 2000.
[49] Matthias Lanzinger and Pablo Barceló. On the power of the Weisfeiler-Leman test for graph motif parameters. *CoRR*, abs/2309.17053, 2023.
[50] Dániel Marx. Approximating fractional hypertree width. *ACM Trans. Algorithms*, 6(2):29:1–29:17, 2010.
[51] Dániel Marx. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. *Journal of the ACM*, 60(6), 2013. Article No. 42.
[52] Stefan Mengel. A short note on the counting complexity of conjunctive queries. *CoRR*, abs/2112.01108, 2021.
[53] Carl A. Miller. Evasiveness of graph properties and topological fixed-point theorems. *Found. Trends Theor. Comput. Sci.*, 7(4):337–415, 2013.
[54] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4602–4609. AAAI Press, 2019.
[55] Daniel Neuen. Homomorphism-distinguishing closedness for graphs of bounded tree-width. *CoRR*, abs/2304.07011, 2023.
[56] Reinhard Pichler and Sebastian Skritek. Tractable counting of the answers to conjunctive queries. *J. Comput. Syst. Sci.*, 79(6):984–1001, 2013.
[57] Marc Roth and Johannes Schmitt. Counting Induced Subgraphs: A Topological Approach to #W[1]-hardness. *Algorithmica*, 82(8):2267–2291, 2020.
[58] Bjarke Hammersholt Roune and Eduardo Sáenz-de-Cabezón. Complexity and algorithms for Euler characteristic of simplicial complexes. *J. Symb. Comput.*, 50:170–196, 2013.
[59] Robert Endre Tarjan and Mihalis Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM J. Comput.*, 13(3):566–579, 1984.
[60] Moshe Y. Vardi. Constraint satisfaction and database theory: a tutorial. In Victor Vianu and Georg Gottlob, editors, *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, May 15-17, 2000, Dallas, Texas, USA*, pages 76–85. ACM, 2000.
[61] Mihalis Yannakakis. Algorithms for acyclic database schemes. In *Very Large Data Bases, 7th International Conference, September 9-11, 1981, Cannes, France, Proceedings*, pages 82–94. IEEE Computer Society, 1981.

A FURTHER PRELIMINARIES

A.1 Parameterised Complexity Theory

A *parameterised counting problem* is a pair consisting of a function $P : \{0,1\}^* \to \mathbb{N}$ and a computable[4] parameterisation $\kappa : \{0,1\}^* \to \mathbb{N}$. For example, in the problem #Clique the function maps an input (a graph G and a positive integer k, encoded as a string in $\{0,1\}^*$ to the number of k-cliques in G. The parameter is K so $\kappa(G,k) = k$.

A parameterised counting problem (P,κ) is called *fixed-parameter tractable* (FPT) if there is a computable function f and an algorithm $\mathbb{A}$ that, given input x, computes $P(x)$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$. We call $\mathbb{A}$ an *FPT-algorithm* for (P,κ).

A *parameterised Turing-reduction* from (P,κ) to (P',κ') is an algorithm $\mathbb{A}$ equipped with oracle access to P' that satisfies the following two constraints: (I) $\mathbb{A}$ is an FPT-algorithm for (P,κ), and (II) there is a computable function g such that, when the algorithm $\mathbb{A}$ is run with input x, every oracle query y to (P',κ') has the property that the parameter $\kappa'(y)$ is bounded by $g(\kappa(x))$. We write $(P,\kappa) \le^{\mathsf{FPT}} (P'\kappa')$ if a parameterised Turing-reduction exists.

Evidence for the non-existence of FPT algorithms is usually given by hardness for the parameterised classes #W[1] and W[1], which can be considered to be the parameterised versions of #P and NP. The definition of those classes uses bounded-weft circuits, and we refer the interested reader e.g. to the standard textbook of Flum and Grohe [34] for a comprehensive introduction. For this work, it suffices to rely on the clique problem to establish hardness for those classes: A parameterised counting problem (P,κ) is #W[1]*-hard* if #Clique $\le^{\mathsf{FPT}} (P,\kappa)$, and it is W[1]*-hard* if Clique $\le^{\mathsf{FPT}} (P,\kappa)$, where Clique is the decision version of #Clique, that is, given G and k, the task is to decide whether there is at least one k-clique in G. As observed in previous works [22], if all variables are existentially quantified, the problem of counting answers to a conjunctive query actually encodes a decision problem. So it comes to no surprise that both complexity classes W[1] and #W[1] are relevant for its classification. It is well known (see e.g. [24, 25, 28] that W[1]-hard and #W[1]-hard problems are not fixed-parameter tractable.

A.2 Signatures and Structures

A *signature* is a finite tuple $\tau = (R_1,\dots,R_s)$ where each R_i is a relation symbol and comes with an arity a_i. The arity of a signature is the maximum arity of its relation symbols. A *structure* $\mathcal{A}$ over τ consists of a finite universe $U(\mathcal{A})$ and a relation $R_i^{\mathcal{A}}$ of arity a_i for each relation symbol R_i of τ. As usual in relational algebra, we view *databases* as relational structures. We encode a structure by listing its signature, its universe and its relations. Therefore, given a structure $\mathcal{A}$ over τ, we set $|\mathcal{A}| = |\tau| + |U(\mathcal{A})| + \sum_{R\in\tau} |R^{\mathcal{A}}| \cdot a_R$, where a_R is the arity of R.

For example, a *graph* G is a structure over the signature (E) where E has arity 2. The *Gaifman graph* of a structure $\mathcal{A}$ has as vertices the universe $U(\mathcal{A})$ of $\mathcal{A}$, and for each pair of vertices u,v, there is an edge $\{u,v\}$ in E if and only if at least one of the relations of $\mathcal{A}$ contains a tuple containing both u and v. Note that the edge set E of a Gaifman graph is symmetric and irreflexive.

The treewidth of graphs and structures is defined as follows

Definition 33 (Tree decompositions, treewidth)**.** Let G be a graph. A *tree decomposition* of G is a pair (T,B), where T is a (rooted) tree, and B assigns each vertex $t \in V(T)$ a *bag* B_t such that the following constraints are satisfied:

(C1) $V(G) = \bigcup_{t\in V(T)} B_t$,
(C2) For each edge $e \in E(G)$ there exists $t \in V(T)$ such that $e \subseteq B_t$, and
(C3) For each $v \in V(G)$, the subgraph of T containing all vertices t with $v \in B_t$ is connected.

The *width* of a tree decomposition is $\max_{t\in V(T)} |B_t| - 1$, and the *treewidth* of G is the minimum width of any tree decomposition of G. Finally, the treewidth of a structure is the treewidth of its Gaifman graph.

Let $\mathcal{A}$ and $\mathcal{B}$ be structures over the same signature τ. Then $\mathcal{A}$ is a *substructure* of $\mathcal{B}$ if $U(\mathcal{A}) \subseteq U(\mathcal{B})$ and, for each relation symbol R in τ, it holds that $R^{\mathcal{A}} \subseteq R^{\mathcal{B}} \cap U(\mathcal{A})^a$, where a is the arity of R. A substructure is *induced* if, for each relation symbol R in τ, we have $R^{\mathcal{A}} = R^{\mathcal{B}} \cap U(\mathcal{A})^a$. A substructure $\mathcal{A}$ of $\mathcal{B}$ with $\mathcal{A} \neq \mathcal{B}$ is a *proper substructure* of $\mathcal{B}$. We also define the *union* $\mathcal{A} \cup \mathcal{B}$ of two structures $\mathcal{A}$ and $\mathcal{B}$ as the structure over τ with universe $U(\mathcal{A}) \cup U(\mathcal{B})$ and $R^{\mathcal{A}\cup\mathcal{B}} = R^{\mathcal{A}} \cup R^{\mathcal{B}}$. Note that the union is well-defined even if the universes are not disjoint.

A.3 Proof of Lemma 23

Proof. By inclusion-exclusion and Observation 21,

$$\begin{aligned}\mathsf{ans}(\Psi \to \mathcal{D}) &= \sum_{\emptyset\neq J\subseteq[\ell]} (-1)^{|J|+1} \cdot \left|\{a : X \to U(\mathcal{D}) \mid \forall j \in J : a \in \mathsf{Ans}((A_j, X) \to \mathcal{D})\}\right| \\ &= \sum_{\emptyset\neq J\subseteq[\ell]} (-1)^{|J|+1} \cdot \mathsf{ans}(\wedge\left(\Psi|_J\right) \to \mathcal{D}).\end{aligned}$$

The claim then follows by collecting #equivalent terms. □

[4] Some authors require the parameterisation to be polynomial-time computable; see the discussion in the standard textbook of Flum and Grohe [34]. In this work, the parameter will always be the size of the input query, which can clearly be computed in polynomial time.

A.4 Isolating Terms in the Homomorphism Basis

We provide the details on how Corollaries 25 and 26 are derived from the work of Chen and Mengel [23]. The following theorem follows from the arguments made in [23, Section 5]. Since it is not stated explicitly, we include a proof for reasons of self-containment.

Theorem 34 (Implicitly by [23])**.** *There is an algorithm $\mathbb{A}$ with the following properties:*

(1) The input of $\mathbb{A}$ is a UCQ Ψ and a database $\mathcal{D}$.
(2) $\mathbb{A}$ has oracle access to the function $\mathcal{D}' \mapsto \mathsf{ans}(\Psi \to \mathcal{D}')$.
(3) The output of $\mathbb{A}$ is a list with entries $((\mathcal{A}, X), \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D}))$ for each $(\mathcal{A}, X)$ in the support of c_Ψ.
(4) $\mathbb{A}$ runs in time $f(|\Psi|) \cdot O(|\mathcal{D}|)$ for some computable function f.

Proof. A crucial operation in the construction is the Tensor product of relational structures. Let $\mathcal{A}$ and $\mathcal{B}$ be structures over the signatures τ_A and τ_B. The structure $\mathcal{A} \otimes \mathcal{B}$ is defined as follows: The signature is $\tau_A \cap \tau_B$, and the universe is $U(\mathcal{A}) \times U(\mathcal{B})$. Moreover, for every relation symbol $R \in \tau_A \cap \tau_B$ with arity r, a tuple $((u_1, v_1), \dots, (u_r, v_r))$ is contained in $R^{\mathcal{A}\otimes\mathcal{B}}$ if and only if $(u_1, \dots, u_r) \in R^{\mathcal{A}}$ and $(v_1, \dots, v_r) \in R^{\mathcal{B}}$.

Observe that $\mathcal{A} \otimes \mathcal{B}$ is of size bounded by and can be computed in time $O(|\mathcal{A}||\mathcal{B}|)$.[5] The algorithm $\mathbb{A}$ proceeds as follows. Let $\Psi = ((\mathcal{A}_1, \dots, \mathcal{A}_\ell), X)$ be the input. For a selected set of structures $\mathcal{B}_1 \dots, \mathcal{B}_k$, specified momentarily, the algorithm queries the oracle on the Tensor products $\mathcal{D} \otimes \mathcal{B}_i$. Using Lemma 23, this yields the following equations:

$$\mathsf{ans}(\Psi \to \mathcal{D} \otimes \mathcal{B}_i) = \sum_{(\mathcal{A},X)} c_\Psi(\mathcal{A}, X) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D} \otimes \mathcal{B}_i).$$

Next, we use the fact (see e.g. [23]) that the Tensor product is multiplicative with respect to counting answers to conjunctive queries:

$$\mathsf{ans}((\mathcal{A}, X) \to \mathcal{D} \otimes \mathcal{B}_i) = \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D}) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{B}_i).$$

In combination, the previous equations yield a system of linear equations:

$$\mathsf{ans}(\Psi \to \mathcal{D} \otimes \mathcal{B}_i) = \sum_{(\mathcal{A},X)} c_\Psi(\mathcal{A}, X) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D}) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{B}_i),$$

the unknowns of which are $c_\Psi(\mathcal{A}, X) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$. Finally, it was shown in [23] and [31] that it is always possible to find $\mathcal{B}_i$ for which the system is non-singular. Moreover, the time it takes to find the $\mathcal{B}_i$ only depends on Ψ. Finally, solving the system yields the terms $c_\Psi(\mathcal{A}, X) \cdot \mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$ from which we can recover $\mathsf{ans}((\mathcal{A}, X) \to \mathcal{D})$ by dividing by $c_\Psi(\mathcal{A}, X)$. It can easily be observed that the overall running time is bounded by $f(|\Psi|) \cdot O(|\mathcal{D}|)$ for some computable function f, as required, which concludes the proof. □

Now note that, using Lemma 23, Corollaries 25 and 26 follow immediately from the previous Theorem.

B OMITTED PROOFS FROM SECTION 3

Proof of Lemma 29. By [23] (see Lemma 48 in the full version of [31] for an explicit statement), there are surjective functions $s : X \to X'$ and $s' : X' \to X$ and homomorphisms $h \in \mathsf{Hom}(\mathcal{A} \to \mathcal{A}')$ and $h' \in \mathsf{Hom}(\mathcal{A}' \to \mathcal{A})$ such that $h|_X = s$ and $h'|_{X'} = s'$.

Clearly, s and s' are bijective. Let $e = \{u, v\}$ be an edge of $G[X]$. Then there exists a tuple $\vec{t}$ of elements of $U(\mathcal{A})$ such that

(i) $\vec{t} \in R^{\mathcal{A}}$ for some relation (symbol) R of the signature of $\mathcal{A}$, and
(ii) u and v are elements of $\vec{t}$.

Since h is a homomorphism, $h(\vec{t}) \in R^{\mathcal{B}}$. Thus $\{h(u), h(v)\} = \{s(u), s(v)\}$ is an edge of $G'[X']$. The backward direction is analogous. □

Proof of Lemma 30 . Clearly, $(\mathcal{A}, X)$ and $(\mathcal{A}', X)$ are #equivalent. Thus it remains to show that $(\mathcal{A}', X)$ is #minimal. Assume for contradiction by Observation 15 that $(\mathcal{A}', X)$ has an #equivalent substructure $\hat{\mathcal{A}}$ that is induced by a set U with $X \subseteq U \subset U(\mathcal{A}')$.

Since $\mathcal{A}'$ is self-join-free and it does not have isolated variables, there is a relation (symbol) R such that $R^{\mathcal{A}'}$ contains precisely one tuple, and $R^{\hat{\mathcal{A}}}$ is empty. Thus, there is no homomorphism from $\mathcal{A}'$ to $\hat{\mathcal{A}}$ and, consequently, $(\hat{\mathcal{A}}, X)$ and $(\mathcal{A}', X)$ cannot be #equivalent, yielding a contradiction and concluding the proof. □

C THE META COMPLEXITY OF COUNTING ANSWERS TO UCQS

We consider the meta-complexity question of deciding whether it is possible to count the answers of a given UCQ in linear time. As pointed out in the introduction, this problem is immediately NP-hard even when restricted to conjunctive queries if we were to allow quantified variables. Therefore, we consider quantifier-free UCQs in this section. Recall the definition of Meta from Section 1.

Name: Meta
Input: A union Ψ of quantifier-free conjunctive queries.
Output: Is it possible to count answers to Ψ in time linear in the size of $\mathcal{D}$, i.e., can the function $\mathcal{D} \mapsto \mathsf{ans}(\Psi \to \mathcal{D})$ be computed in time $O(|\mathcal{D}|)$.

[5] Compute the Cartesian product of $U(\mathcal{A})$ and $U(\mathcal{B})$ and then, for every relation $R \in \tau_A \cap \tau_B$ iterate over all pairs of tuples in $R^{\mathcal{A}}$ and $R^{\mathcal{B}}$ and add their point-wise product to $R^{\mathcal{A}\otimes\mathcal{B}}$.

Since we focus in this section solely on quantifier-free queries, it will be convenient to simplify our notation as follows. As all variables are free, we will identify a conjunctive query φ just by its associated structure, that is, we will write $\varphi = \mathcal{A}$, rather than $\varphi = (\mathcal{A}, U(\mathcal{A}))$. Similarly, we represent a union of quantifier-free conjunctive queries Ψ as a tuple of structures $\Psi = (\mathcal{A}_1, \dots, \mathcal{A}_\ell)$.

For studying the complexity of Meta, it will be crucial to revisit the classification of linear-time counting of answers to quantifier-free conjunctive queries: The following theorem is well known and was discovered multiple times by different authors in different contexts.[6] This is Theorem 4 from the introduction, which we now restate in a version that expresses CQs as structures.

Theorem 35 (See Theorem 12 in [17], and [7, 8, 11]). *Let $\mathcal{A}$ be a quantifier-free conjunctive query and suppose that the Triangle Conjecture is true. Then the function $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{A} \to \mathcal{D})$ is computable in linear time if and only if $\mathcal{A}$ is acyclic.*

Theorem 35 yields an efficient way to check whether counting answers to a quantifier-free conjunctive query φ can be done in linear time: Just check whether φ is acyclic. In stark contrast, we show that no easy criterion for linear time tractability of counting answers to *unions* of conjunctive queries is possible, unless some conjectures of fine-grained complexity theory fail. In fact, our Theorem 5, which we restate here for convenience, precisely determines the complexity of Meta.

Theorem 5. *Meta can be solved in time $2^{O(\ell)} \cdot |\Psi|^{\mathrm{poly}(\log|\Psi|)}$, where ℓ is the number of conjunctive queries in the union, if the Triangle Conjecture is true. Moreover,*

- *If the Triangle Conjecture is true then Meta is NP-hard. If, additionally, ETH is true, then Meta cannot be solved in time $2^{o(\ell)}$.*
- *If SETH is true then Meta is NP-hard and cannot be solved in time $2^{o(\ell)}$.*
- *If the non-uniform ETH is true then Meta is NP-hard and Meta $\notin \bigcap_{\varepsilon>0} \mathrm{DTime}(2^{\varepsilon \cdot \ell})$.*

The lower bounds remain true even if Ψ is a union of self-join-free and acyclic conjunctive queries over a binary signature (that is, of arity 2).

The lower bounds in Theorem 5 imply that the exponential dependence on ℓ in our $2^{O(\ell)} \cdot |\Psi|^{\mathrm{poly}(\log|\Psi|)}$ time algorithm for Meta cannot be significantly improved, unless standard assumptions fail.

The remainder of this section is devoted to the proof of Theorem 5. It is split into two parts: In the first and easier part (Section C.1), we construct the algorithm for Meta. For this, all we need to do is to translate the problem into the homomorphism basis (see Section 2.3), i.e., we transform the problem of counting answers to Ψ into the problem of evaluating a linear combination of terms, each of which can be determined by counting the answers to a conjunctive query. This is done in Lemma 36. The second part (Section C.2) concerns the lower bounds and is more challenging. The overall strategy is as follows: In the first step, we use a parsimonious reduction from 3-SAT to computing the reduced Euler Characteristic of a complex. The parsimonious reduction is due to Roune and Sáenz-de-Cabezón [58]. In combination with the Sparsification Lemma [44], this reduction becomes tight enough for our purposes. In the second step, we show how to encode a complex Δ into a union of acyclic conjunctive queries Ψ such that the following is true: The reduced Euler Characteristic of Δ is zero if and only if all terms in the homomorphism basis are acyclic. The lower bound results of Theorem 5 are established in Lemmas 49, 50, and 51.

C.1 Solving Meta via Inclusion-Exclusion

Lemma 36. *The problem Meta can be solved in time $|\Psi|^{O(\log|\Psi|)} \cdot 2^{O(\ell)}$, where ℓ is the number of conjunctive queries in the union, if the Triangle Conjecture is true.*

Proof. If the Triangle Conjecture is true, then Theorem 35 implies that counting answers to a quantifier-free conjunctive query is solvable in linear time if and only if the query is acyclic. In combination with Corollary 25 we obtain that counting answers to a UCQ $\Psi = (\mathcal{A}_1, \dots, \mathcal{A}_\ell)$ can be done in linear time if and only if each $\mathcal{A}$ with $c_\Psi(\mathcal{A}) \neq 0$ is acyclic.

Recall that #equivalence is the same as isomorphism for quantifier-free queries (Definition 14). By Definition 22,

$$c_\Psi(\mathcal{A}) = \sum_{\substack{J \subseteq [\ell] \\ \wedge(\Psi|_J) \cong \mathcal{A}}} (-1)^{|J|+1}.$$

This suggests the following algorithm for Meta with input Ψ. For each subset $J \subseteq [\ell]$, compute $\wedge(\Psi|_J)$. Afterwards, using Babai's algorithm [6] collect the isomorphic terms and compute $c_\Psi(\wedge(\Psi|_J))$ for each $J \subseteq [\ell]$ in time $2^{O(\ell)} \cdot |\Psi|^{\mathrm{poly}(\log|\Psi|)}$. Clearly, $c_\Psi(\mathcal{A}) = 0$ for every $\mathcal{A}$ that is not isomorphic to any $\wedge(\Psi|_J)$.

Finally, output 1 if each $\mathcal{A}$ with $c_\Psi(\mathcal{A}) \neq 0$ is acyclic (each of these checks can be done in linear time [59]), and output 0 otherwise. The total running time of this algorithm is bounded from above by $2^{O(\ell)} \cdot |\Psi|^{\mathrm{poly}(\log|\Psi|)}$. ◻

C.2 Fine-grained Lower bounds for Meta

For our hardness proof for Meta, we will construct a reduction from the computation of the reduced Euler characteristic of an abstract simplicial complex. We begin by introducing some central notions about (abstract) simplicial complexes.

[6] We remark that [11, Theorem 7] focuses on the special case of graphs and *near* linear time algorithms. However, in the word RAM model with $O(\log n)$ bits, a linear time algorithm is possible [20].

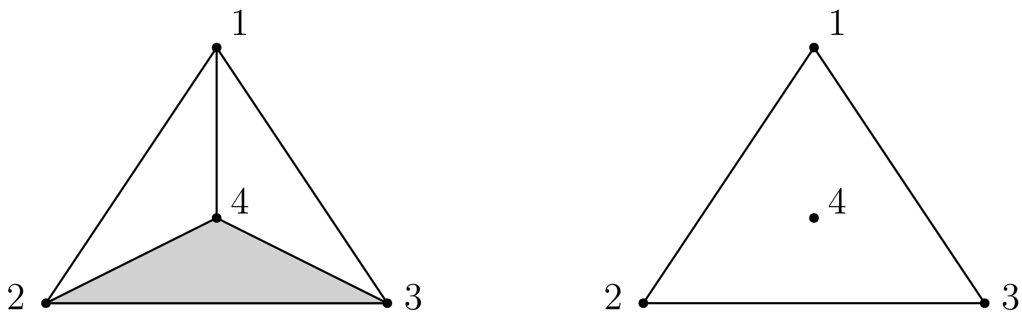

Figure 1: Two complexes over the groundset $\Omega = \{1,2,3,4\}$. Let Δ_1 be the complex shown on the left. It has facets $\{2,3,4\}$, $\{1,2\}$, $\{1,3\}$, and $\{1,4\}$. Let Δ_2 be the complex shown on the right, with facets $\{1,2\}$, $\{2,3\}$, $\{1,3\}$, and $\{4\}$. The reduced Euler characteristic of these complexes is computed as follows: Since Δ_1 has one face of size 3 ($\{2,3,4\}$), 6 faces of size 2, 4 faces of size 1, and the empty set as face of size 0 it holds that $\hat{\chi}(\Delta_1) = -(-1+6-4+1) = -2$. Similarly, we have $\hat{\chi}(\Delta_2) = -(3-4+1) = 0$.

C.2.1 Simplicial Complexes. A simplicial complex captures the geometric notion of an independence system. We will use the corresponding combinatorial description, which is also known as *abstract simplicial complex*, and defined as follows.

Definition 37. A *complex* (short for abstract simplicial complex) Δ is a pair consisting of a nonempty finite ground set Ω and a set of *faces* $\mathcal{I} \subseteq 2^{\Omega}$ that includes all singletons and is a downset. That is, the set of faces $\mathcal{I}$ satisfies the following criteria.

- $\forall S \subseteq \Omega : S \in \mathcal{I} \Rightarrow \forall S' \subseteq S : S' \in \mathcal{I}$, and
- $\forall x \in \Omega : \{x\} \in \mathcal{I}$.

The inclusion-maximal faces in $\mathcal{I}$ are called *facets*. Unless stated otherwise, we encode complexes by the ground set Ω and the set of facets. Then $|\Delta|$ is its encoding length.

Definition 38. The *reduced Euler characteristic* of a complex $\Delta = (\Omega, \mathcal{I})$ is defined as

$$\hat{\chi}(\Delta) := -\sum_{S \in \mathcal{I}} (-1)^{|S|}\,.$$

Consider for example the complexes Δ_1 and Δ_2 in Figure 1, given with a computation of their reduced Euler characteristic.

Let $\Delta = (\Omega, \mathcal{I})$ be a complex. Consider distinct elements x and y in Ω. We say that x *dominates* y if, for every $S \in \mathcal{I}$, $y \in S$ implies $S \cup \{x\} \in \mathcal{I}$. For example, x and y dominate each other if they are contained in the same facets. We start with a simple observation:

Lemma 39. *Let $\Delta = (\Omega, \mathcal{I})$ be a complex and let $x, y \in \Omega$. Then x dominates y if and only if each facet that contains y must also contain x.*

Proof. For the forward direction suppose x dominates y. Let F be a facet that contains y. Then $F \cup \{x\} \in \mathcal{I}$. Since facets are inclusion maximal in $\mathcal{I}$ it follows that $F = F \cup \{x\}$, that is, $x \in F$.

For the backward direction suppose that each facet containing y must also contain x. Let $S \in \mathcal{I}$ with $y \in S$. Then there is a facet F with $S \subseteq F$. Since $x \in F$ we have $S \cup \{x\} \subseteq F$ and hence $S \cup \{x\} \in \mathcal{I}$. Therefore x dominates y. □

We say that a complex $\Delta = (\Omega, \mathcal{I})$ is *irreducible* if, for every $y \in \Omega$, there is no $x \in \Omega \setminus \{y\}$ that dominates y.

Given $\Delta = (\Omega, \mathcal{I})$ and $y \in \Omega$, we define $\Delta \setminus y$ to be the complex obtained from Δ by deleting all faces that contain y and by deleting y from Ω. The following lemma seems to be folklore. We include a proof only for reasons of self-containment.

Lemma 40. *Let $\Delta = (\Omega, \mathcal{I})$ be a complex. If $y \in \Omega$ is dominated by some $x \in \Omega \setminus \{y\}$ then $\hat{\chi}(\Delta) = \hat{\chi}(\Delta \setminus y)$.*

Proof. Let $\Delta = (\Omega, \mathcal{I})$. Write $\mathcal{I}_y$ for the set of all faces containing y. Consider the mapping $b : \mathcal{I}_y \to \mathcal{I}_y$

$$b(S) := \begin{cases} S \cup \{x\} & x \notin S \\ S \setminus \{x\} & x \in S\,. \end{cases}$$

Note that b is well-defined since $S \cup \{x\} \in \mathcal{I}_y$ because $y \in S$ and x dominates y. Observe that b induces a partition of $\mathcal{I}_y$ in pairs $\{S, S \cup \{x\}\}$ for $x \notin S$. For those pairs, we clearly have $|S| + 1 = |S \cup \{x\}|$. Thus

$$\sum_{S \in \mathcal{I}_y} (-1)^{|S|} = 0\,,$$

and therefore

$$\hat{\chi}(\Delta) = -\sum_{S \in \mathcal{I}} (-1)^{|S|} = -\sum_{S \in \mathcal{I} \setminus \mathcal{I}_y} (-1)^{|S|} = \hat{\chi}(\Delta \setminus y)\,.$$

□

Definition 41. Two complexes $\Delta_1 = (\Omega_1, \mathcal{I}_1)$ and $\Delta_2 = (\Omega_2, \mathcal{I}_2)$ are *isomorphic* if there is a bijection $b : \Omega_1 \to \Omega_2$ such that $S \in \mathcal{I}_1$ if and only if $b(S) \in \mathcal{I}_2$ for each $S \subseteq \Omega$, where $b(S) = \{b(x) \mid x \in S\}$.

Finally, a complex $(\Omega, \mathcal{I})$ is called *trivial* if it is isomorphic to $(\{x\}, \{\emptyset, \{x\}\})$.

C.2.2 The Main Reduction. To begin with, we require a conjunctive query whose answers cannot be counted in linear time under standard assumptions. To this end, we define, for positive integers k and t, a binary relational structure $\mathcal{K}_t^k$ as follows. Start with a t-clique K_t and k-stretch every edge, that is, each edge of K_t is replaced by a path consisting of k edges. We denote the resulting graph by K_t^k. For each edge e of K_t^k, we introduce a relation $R_e = \{e\}$ of arity 2. The structure $\mathcal{K}_t^k$ has universe $V(K_t^k)$ and relations $(R_e)_{e \in E(K_t^k)}$.

Observation 42. *Let k and t be positive integers. The structure $\mathcal{K}_t^k$ is self-join-free and has arity* 2.

Lemma 43. *Suppose that the non-uniform ETH holds. For all positive integers d, there is a t such that for each k, the function $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{K}_t^k \to \mathcal{D})$ cannot be computed in time $O(|\mathcal{D}|^d)$.*

Proof. We again write K_t for the t-clique and K_t^k for the k-stretch of K_t. Chen, Eickmeyer and Flum [26] proved that, under non-uniform ETH, for each positive integer d, there is a t such that determining whether a graph G contains K_t as a subgraph cannot be done in time $O(|G|^d)$.

We construct a simple linear-time reduction to computing $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{K}_t^k \to \mathcal{D})$. Since for each edge e of the underlying graph K_t^k, the structure $\mathcal{K}_t^k$ has a separate binary single-element relation R_e, the input database $\mathcal{D}$ (of the same signature as $\mathcal{K}_t^k$) also has a relation R'_e whose elements can be interpreted as e-coloured edges. This means that the problem of computing $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{K}_t^k \to \mathcal{D})$ can equivalently be expressed as the following problem: Given a graph G', that comes with an edge-colouring $\mathsf{col} : E(G') \mapsto E(K_t^k)$, count the homomorphisms h from K_t^k to G' such that for each edge e of K_t^k we have $\mathsf{col}(h(e)) = e$.

We now reduce the problem of determining whether a graph G contains K_t to this problem. Given input G, we construct an edge-coloured graph G' as follows. Each edge of G is replaced by $\binom{t}{2}$ paths of length k, and we colour the edges of the i-th of those paths with the k edges of the k-stretch of the i-th edge e_i of K_t. It is easy to see that G contains a t-clique if and only if there is at least one homomorphism from K_t^k to G' that preserves the edge-colours. Moreover, the construction of G' can clearly be done in linear time. □

Before proving Theorem 5, we need to take a detour to examine the complexity of computing the reduced Euler Characteristic of a complex associated with a UCQ. To begin with, we introduce the notion of a "power complex".

Definition 44. Let $\mathcal{U}$ be a finite set, and let $\Omega \subseteq 2^{\mathcal{U}}$ be a set system that does not contain $\mathcal{U}$. The *power complex* $\Delta_{\Omega,\mathcal{U}}$ is a complex with ground set Ω and faces

$$\mathcal{I} = \left\{ S \subseteq \Omega \;:\; \bigcup_{A \in S} A \neq \mathcal{U} \right\}.$$

It is easy to check that $\Delta_{\Omega,\mathcal{U}}$ is a complex — for each $x \in \Omega$, the set $\{x\}$ is in $\mathcal{I}$ since Ω does not contain $\mathcal{U}$. Although we typically encode complexes by the ground set Ω and the set of facets, in the case of a power complex we list the elements of $\mathcal{U}$ and Ω.

In the first step, we show that each complex is isomorphic to a power complex.

Lemma 45. *Let $\Delta = (\Omega, \mathcal{I})$ be a non-trivial irreducible complex. It is possible to compute, in polynomial time in $|\Delta|$, a set $\mathcal{U}$ and a set $\hat{\Omega} \subseteq 2^{\mathcal{U}}$ with $\mathcal{U} \notin \hat{\Omega}$ such that Δ is isomorphic to the power complex $\Delta_{\hat{\Omega},\mathcal{U}}$.*

Proof. Let $F_1, \dots, F_k$ be the facets of Δ so that the encoding of Δ consists of Ω and $F_1, \dots, F_k$, and $|\Delta|$ is the length of this encoding. For each $i \in [k]$, we introduce an element E_i corresponding to F_i. Let $\mathcal{U} = \{E_1, \dots, E_k\}$. Next, we define a mapping $b : \Omega \to 2^{\mathcal{U}}$ as follows:

$$b(x) := \{E_i \mid x \notin F_i\}.$$

Observe that b is injective since otherwise two elements of Ω are contained in the same facets of Δ, which means that they dominate each other. Also, note that $b(x) = \mathcal{U}$ implies that x is contained in every facet of Δ and therefore dominates all other elements in Ω. This gives a contradiction as Δ is non-trivial and therefore Ω contains at least two elements (none of which dominate each other).

We choose $\hat{\Omega}$ as the range of b. Then, clearly, b is a bijection from Ω to $\hat{\Omega}$. Furthermore, $\mathcal{U}$ and $\hat{\Omega}$ can be constructed in time polynomial in $|\Delta|$.

It remains to prove that b is also an isomorphism from Δ to $\Delta_{\hat{\Omega},\mathcal{U}}$. To this end, we need to show that

$$S \in \mathcal{I} \Leftrightarrow \bigcup_{A \in b(S)} A \neq \mathcal{U},$$

where $b(S) = \{b(x) \mid x \in S\}$.

For the first direction, let $S \in \mathcal{I}$. W.l.o.g. we have $S \subseteq F_1$. Then, for all $x \in S$, $E_1 \notin b(x)$. Hence $E_1 \notin \bigcup_{A \in b(S)} A$. For the other direction, suppose that $S \notin \mathcal{I}$. Then S is not a subset of any facet. Consequently, for every $i \in [k]$, there exists $x_i \in S$ such that $x_i \notin F_i$. By definition of b, we thus have that, for every $i \in [k]$, there exists $x_i \in S$ such that $E_i \in b(x_i)$. Thus $\bigcup_{A \in b(S)} A = \mathcal{U}$. □

Recall, for example, the complex Δ_1 in Figure 1. Δ_1 has facets $F_1 = \{2,3,4\}$, $F_2 = \{1,2\}$, $F_3 = \{1,3\}$, and $F_4 = \{1,4\}$. Since Δ_1 is irreducible and non-trivial, we can apply the construction in the previous lemma to construct an isomorphic power complex: We set $\mathcal{U} = \{E_1, E_2, E_3, E_4\}$, since Δ_1 has 4 facets. Moreover, the ground set $\hat{\Omega}$ of the power complex is the range of the function $b : \Omega \to 2^{\mathcal{U}}$ with $b(x) := \{E_i \mid x \notin F_i\}$, that is, $b(1) = \{E_1\}$ since $1 \notin F_1$, and similarly, $b(2) = \{E_3, E_4\}$, $b(3) = \{E_2, E_4\}$, and $b(4) = \{E_2, E_3\}$. Then the facets $\hat{\mathcal{I}}$ of the power complex are $\{E_1, E_3, E_4\}$, $\{E_1, E_2, E_4\}$, $\{E_1, E_2, E_3\}$, and $\{E_2, E_3, E_4\}$. Then Δ_1 and the power complex $\Delta_{\hat{\Omega},\mathcal{U}}$ are isomorphic, where for instance $F_2 = \{1,2\}$ corresponds to $b(1) \cup b(2) = \{E_1, E_3, E_4\}$. When applied to Δ_2, the same construction yields a power complex with $\mathcal{U} = \{E_1, E_2, E_3, E_4\}$ and ground set $\{\{E_3, E_4\}, \{E_1, E_4\}, \{E_2, E_4\}, \{E_1, E_2, E_3\}\}$.

We now introduce the main technical result that we use to establish lower bounds for Meta; note that computability result in the last part of Lemma 46 will just be required for the construction of exemplary classes of UCQs in the appendix.

Lemma 46. *For each positive integer t, there is a polynomial-time algorithm $\hat{\mathbb{A}}_t$ that, when given as input a non-trivial irreducible complex $\Delta = (\Omega, \mathcal{I})$ with $\Omega \notin \mathcal{I}$, computes a union of quantifier-free conjunctive queries $\Psi = (\mathcal{B}_1, \dots, \mathcal{B}_\ell)$ satisfying the following constraints:*

(1) $\wedge(\Psi) \cong \mathcal{K}_t^k$ for some $k \geq 1$.
(2) $c_\Psi(\wedge(\Psi)) = -\hat{\chi}(\Delta)$.
(3) For all relational structures $\mathcal{B} \ncong \wedge(\Psi)$, $c_\Psi(\mathcal{B}) \neq 0$ implies that $\mathcal{B}$ is acyclic.
(4) $\ell \leq |\Omega|$.
(5) For all $i \in [\ell]$ the conjunctive query $\mathcal{B}_i$ is acyclic and self-join-free, and has arity 2.

Moreover, the algorithm $\hat{\mathbb{A}}_t$ can be explicitly constructed from t.

Proof. The algorithm $\hat{\mathbb{A}}_t$ applies Lemma 45 to obtain $\mathcal{U}$ and $\hat{\Omega}$ such that the power-complex $\Delta_{\hat{\Omega},\mathcal{U}}$ is isomorphic to Δ. Let $k = |\mathcal{U}|$ and assume w.l.o.g. that $\mathcal{U} = [k]$. Let $\hat{\Omega} = \{A_1, \dots, A_\ell\}$. Let e^j be the j-th edge of the k-stretch of e in K_t^k and recall that $\mathcal{K}_t^k$ contains the relations $R_{e^i} = \{e^i\}$ for each $e \in E(K_t)$ and $i \in [k]$. For each $i \in [k]$ we define $\mathcal{E}_i$ to be the substructure of $\mathcal{K}_t^k$ containing the universe $V(K_t^k)$ and the relations R_{e^i} (one relation for each $e \in E(K_t)$). The algorithm constructs $\Psi = (\mathcal{B}_1, \dots, \mathcal{B}_\ell)$ as follows: For each $j \in [\ell]$, set $\mathcal{B}_j = \cup_{i \in A_j} \mathcal{E}_i$, that is, $\mathcal{B}_j$ is the substructure of $\mathcal{K}_t^k$ obtained by taking all relations in $\mathcal{E}_i$ for all $i \in A_j$. The pseudocode for $\hat{\mathbb{A}}_t$ is as follows.

Algorithm 1 Pseudocode for $\hat{\mathbb{A}}_t$

```
Input: Δ = (Ω, I)
(F_1, ..., F_k) ← facets of Δ
for x ∈ Ω do
    A_x ← {i | x ∉ F_i}          ▷ U = [k] and Ω̂ = {A_x | x ∈ Ω} = {A_1, ..., A_ℓ} yield the power complex
end for
K_t ← t-clique, m ← binom(t, 2)  ▷ e_i denotes the i-th edge of K_t
K_t^k ← k-stretch of K_t          ▷ e_i^j denotes the j-th edge of the k-stretch of e_i ∈ E(K_t)
for i ∈ [m], j ∈ [k] do
    R_{e_i^j} ← {e_i^j}
end for
for i ∈ [k] do
    E_i ← (V(K_k^t), R_{e_1^i}, ..., R_{e_m^i})
end for
for j ∈ [ℓ] do
    B_j ← ⋃_{i ∈ A_j} E_i
end for
Output: Ψ = (B_1, ..., B_ℓ)
```

We now prove the required properties of Ψ.

(1) Recall that Ω is not a facet. Since Δ and $\Delta_{\hat{\Omega},\mathcal{U}}$ are isomorphic, $\hat{\Omega}$ is not a facet. Thus $\bigcup_{i \in [\ell]} A_i = \mathcal{U}$ by the definition of power complexes. Since $\mathcal{B}_j$ contains all relations in $\mathcal{E}_i$ with $i \in A_j$, we have that $\wedge(\Psi) = \mathcal{K}_t^k$ as desired.
(2) Recall from Definition 22 that $\mathcal{I}(\mathcal{B}, X) = \{J \subseteq [\ell] \mid (\mathcal{B}, X) \sim \wedge(\Psi|_J)\}$. Since we are in the setting of quantifier-free queries ($X = U(\mathcal{B})$; we will drop the X as before), equivalence ($\sim$) is equivalent to isomorphism. Thus, we have that $c_\Psi(\wedge(\Psi))$ is equal to

$$\sum_{\substack{J \subseteq [\ell] \\ \wedge(\Psi) \cong \wedge(\Psi|_J)}} (-1)^{|J|+1} = \sum_{\substack{J \subseteq [\ell] \\ \wedge(\Psi) \ncong \wedge(\Psi|_J)}} (-1)^{|J|} = \sum_{\substack{J \subseteq [\ell] \\ \cup_{j \in J} A_j \neq \mathcal{U}}} (-1)^{|J|} = -\hat{\chi}(\Delta_{\hat{\Omega},\mathcal{U}}) = -\hat{\chi}(\Delta)\,.$$

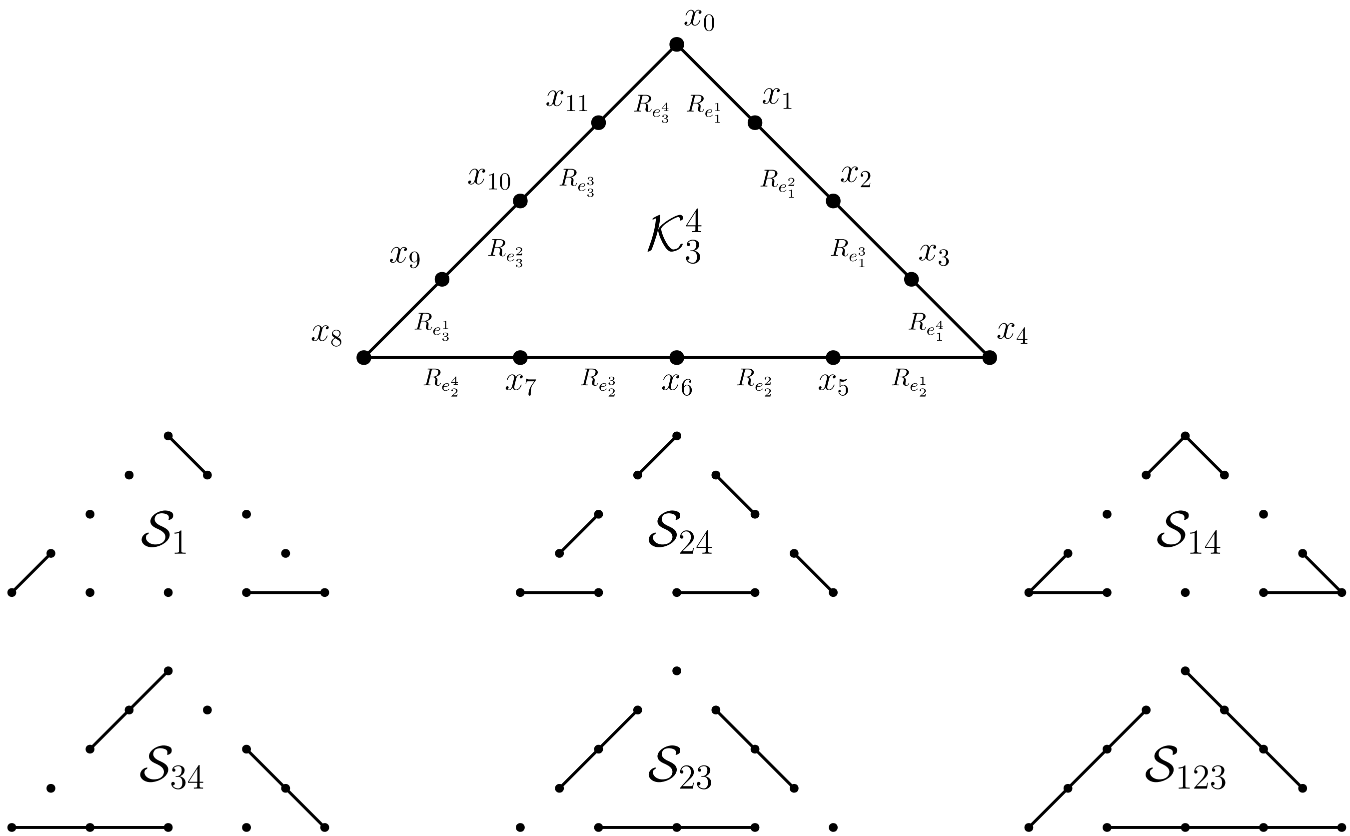

Figure 2: *(Top:)* The structure $\mathcal{K}_3^4$. *(Bottom:)* Substructures $\mathcal{S}_A$ for some selected $A \subseteq [4]$. Observe that all of the $\mathcal{S}_A$ are *acyclic.*

(3) If $\mathcal{B} \not\cong \bigwedge(\Psi)$ and $c_\Psi(\mathcal{B}) \neq 0$, then $\mathcal{B}$ is a substructure of $\mathcal{K}_t^k$ that is missing at least one of the $\mathcal{E}_i$. The claim holds since each $\mathcal{E}_i$ is a feedback edge set (i.e., the deletion of all tuples in $\mathcal{E}_i$ breaks every cycle), because deleting $\mathcal{E}_i$ corresponds to deleting one edge from each stretch.

(4) Follows immediately by construction.

(5) The $\mathcal{B}_i$ are self-join-free and of arity 2 since they are substructures of $\mathcal{K}_t^k$ (see Observation 42). For acyclicity, we use the fact that for each j we have $A_j \neq \mathcal{U}$ by definition of the power complex, which implies the claim by the same argument as 3.

With all cases completed, the proof is concluded. □

To provide a concrete example, we apply the construction in Lemma 46 to the complexes Δ_1 and Δ_2 in Figure 1. For this example, we will choose $t = 3$. Since both Δ_1 and Δ_2 have four facets, we will choose $k = 4$. The structure $\mathcal{K}_t^k$ is depicted in Figure 2, together with six selected substructures corresponding to the $\mathcal{B}_j$ in the proof of the Lemma 46.

Let us start by applying $\hat{\mathbb{A}}_3$ to Δ_1. Recall from the previous example, that the power complex of Δ_1 has ground set $\{A_1, A_2, A_3, A_4\}$ with $A_1 = \{E_1\}$, $A_2 = \{E_3, E_4\}$, $A_3 = \{E_2, E_4\}$, and $A_4 = \{E_2, E_3\}$. Thus $\hat{\mathbb{A}}_3$ returns the UCQ $\Psi_1 = (\mathcal{S}_1, \mathcal{S}_{34}, \mathcal{S}_{24}, \mathcal{S}_{23})$. Similarly, applying $\hat{\mathbb{A}}_3$ to Δ_2 yields the UCQ $\Psi_2 = (\mathcal{S}_{24}, \mathcal{S}_{34}, \mathcal{S}_{14}, \mathcal{S}_{123})$.

For the purpose of illustration, let us write Ψ_1 and Ψ_2 as formulas. To this end, given a non-empty subset A of $\{1, 2, 3, 4\}$, define the conjunctive query φ_A as follows:

$$\varphi_A = \bigwedge_{a \in A} R_{e_1^a}(x_{a-1}, x_a) \wedge R_{e_2^a}(x_{4+a-1}, x_{4+a}) \wedge R_{e_3^a}(x_{8+a-1}, x_{8+a}),$$

where indices are taken modulo 12. Observe that the $\mathcal{S}_A$ depicted in Figure 2 are the structures associated to φ_A for $A \in \{\{1\}, \{2, 4\}, \{1, 4\}, \{3, 4\}, \{2, 3\}, \{1, 2, 3\}\}$. Then

$$\Psi_1(x_0, \ldots, x_{11}) = \varphi_1 \vee \varphi_{34} \vee \varphi_{24} \vee \varphi_{23}, \text{ and}$$
$$\Psi_2(x_0, \ldots, x_{11}) = \varphi_{24} \vee \varphi_{34} \vee \varphi_{14} \vee \varphi_{123}$$

Note that $\bigwedge(\Psi_1) = \bigwedge(\Psi_2) = \mathcal{K}_3^4$. Now recall that $\hat{\chi}(\Delta_1) \neq 0$ and $\chi(\hat{\Delta}_2) = 0$. Thus, by Item 2. of Lemma 46, it holds that $c_{\Psi_1}(\mathcal{K}_3^4) \neq 0$, which means that there is an acyclic CQ in the CQ expansion of Ψ_1. On the contrary, by Item 3. of Lemma 46, for all $(\mathcal{A}, X)$, we have that $c_{\Psi_2}(\mathcal{A}, X) \neq 0$ implies that $(\mathcal{A}, X)$ is acyclic. So, as a conclusion of our example, using complexity monotonicity (Corollary 25) and Theorem 35, we obtain as an immediate consequence:

Corollary 47. *While it is not possible to count answers to Ψ_1 in linear time (in the input database), unless the Triangle Conjecture fails, for Ψ_2 such a linear-time algorithm exists (even though $\bigwedge(\Psi_1) = \bigwedge(\Psi_2)$).*

Proof. By Corollary 25, it is possible to count answers to a UCQ Ψ in linear time if and only if for each #minimal conjunctive query $(\mathcal{A}, X)$ with $c_\Psi(\mathcal{A}, X) \neq 0$, it is possible to count answers to $(\mathcal{A}, X)$ in linear time. Since Ψ_1 and Ψ_2 are quantifier-free, all conjunctive queries $\bigwedge(\Psi_1|_J)$ and $\bigwedge(\Psi_2|_J)$ are also quantifier-free and thus #minimal. For all $(\mathcal{A}, X)$, $c_{\Psi_2}(\mathcal{A}, X) \neq 0$ implies that $(\mathcal{A}, X)$ is acyclic, so it follows from Theorem 35 that it is possible to count answers to Ψ_2 in linear time. Moreover, given that $\mathcal{K}_3^4$ is not acyclic, Theorem 35 also implies that counting answers to $\mathcal{K}_3^4$ cannot be done in linear time, unless the Triangle Conjecture fails. Since $c_{\Psi_1}(\mathcal{K}_3^4) \neq 0$, counting answers to Ψ_1 cannot be done in linear time, unless the Triangle Conjecture fails. □

We will now conclude the hardness proof for Meta.

Lemma 48. *For each positive integer t, there is a polynomial-time algorithm $\mathbb{A}_t$ that, when given as input a complex $\Delta = (\Omega, \mathcal{I})$, either computes $\hat{\chi}(\Delta)$, or computes a union of quantifier-free conjunctive queries $\Psi = (\mathcal{B}_1, \dots, \mathcal{B}_\ell)$ satisfying the following constraints:*

(1) $\bigwedge(\Psi) \cong \mathcal{K}_t^k$ for some $k \geq 1$.
(2) $c_\Psi(\bigwedge(\Psi)) = -\hat{\chi}(\Delta)$.
(3) For all relational structures $\mathcal{B} \not\cong \bigwedge(\Psi)$, $c_\Psi(\mathcal{B}) \neq 0$ implies that $\mathcal{B}$ is acyclic.
(4) $\ell \leq |\Omega|$.
(5) For all $i \in [\ell]$ the conjunctive query $\mathcal{B}_i$ is acyclic and self-join-free, and has arity 2.

Proof. Given $\Delta = (\Omega, \mathcal{I})$, we can successively apply Lemma 40 without changing the reduced Euler characteristic, until the resulting simplicial complex is irreducible. This can be done in polynomial time: By Lemma 39 it suffices to check whether there are $y \neq x \in \Omega$ such that each facet containing y also contains x. If no such pair exists, we are done. Otherwise we delete y from Ω and from all facets and continue recursively. Clearly, the number of recursive steps is bounded by $|\Omega|$ so the run time is at most a polynomial in $|\Delta|$.

If this process makes the complex trivial, we output 0 (i.e., the reduced Euler Characteristic of the trivial complex). We can furthermore assume that Ω is not a facet, i.e., that $\Omega \notin \mathcal{I}$, since in this case every subset of Ω is a face and the reduced Euler Characteristic is 0. We can thus assume that Δ is non-trivial and irreducible, and that Ω is not a facet. Therefore, we can use the algorithm $\hat{\mathbb{A}}_t$ from Lemma 46. This concludes the proof. □

We are now able to prove our lower bounds for Meta.

Lemma 49. *If the Triangle Conjecture is true then Meta is* NP*-hard. If the Triangle Conjecture and ETH are both true then Meta cannot be solved in time $2^{o(\ell)}$ where ℓ is the number of conjunctive queries in its input. Both results remain true even if the input to Meta is restricted to be over a binary signature.*

Proof. For the first result, we assume the triangle conjecture and show that Meta is NP-hard. The input to Meta is a formula Ψ' which is a union of quantifier-free, self-join-free, and acyclic conjunctive queries. The goal is to decide whether counting answers of Ψ' (in an input database) can be done in linear time.

We reduce from 3-SAT. Let F be a 3-SAT formula. The first step of our reduction is to apply a reduction from [58]. Concretely, [58] gives a reduction that, given a 3-SAT formula F with n variables and m clauses, outputs in polynomial time a complex Δ such that F is satisfiable if and only if $\hat{\chi}(\Delta) \neq 0$. Moreover, the ground set of Δ has size $O(n + m)$.

Let $t = 3$ and let $\mathbb{A}_t$ be the algorithm from Lemma 48. Consider running $\mathbb{A}_t$ with input Δ. If $\mathbb{A}_t$ outputs $\hat{\chi}(\Delta)$ then we can check immediately whether $\hat{\chi}(\Delta) = 0$, which determines whether or not F is satisfiable. Otherwise, $\mathbb{A}_t$ outputs a formula Ψ which is a union of self-join-free, quantifier-free, and acylic conjunctive queries of arity 2, and further has the property that $c_\Psi(\bigwedge(\Psi)) = -\hat{\chi}(\Delta)$. Since $\mathbb{A}_t$ is a polynomial-time algorithm, the number of conjunctive queries ℓ in Ψ is at most a polynomial in $n + m$.

We wish to show that determining whether counting answers of Ψ can be done in linear time would reveal whether or not $c_\Psi(\bigwedge(\Psi)) = 0$ (which would in turn reveal whether F is satisfiable).

By Corollary 25, it is possible to compute $\mathcal{D} \mapsto \mathsf{ans}(\Psi \to \mathcal{D})$ in linear time if and only if, for each relational structure $\mathcal{A}$ with $c_\Psi(\mathcal{A}) \neq 0$, the function $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{A} \to \mathcal{D})$ can be computed in linear time.

So the intermediate problem is to check whether, for each relational structure $\mathcal{A}$ with $c_\Psi(\mathcal{A}) \neq 0$, the function $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{A} \to \mathcal{D})$ can be computed in linear time. We wish to show that solving the intermediate problem (in polynomial time) would enable us to determine whether or not $c_\Psi(\bigwedge(\Psi)) = 0$ (also in polynomial time).

Item 1 of Lemma 48 implies that there is a positive integer k such that $\bigwedge(\Psi) \cong \mathcal{K}_3^k$. Thus, $\bigwedge(\Psi)$ is not acyclic. However, Item 3 of Lemma 48 implies that every relational structure $\mathcal{A} \not\cong \bigwedge(\Psi)$ in the intermediate problem, the structure $\mathcal{A}$ is acyclic. Theorem 35 shows (assuming the triangle conjecture) that, for each $\mathcal{A}$, $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{A} \to \mathcal{D})$ can be computed in linear time if and only if $\mathcal{A}$ is acyclic. We conclude that the answer to the intermediate problem is yes iff $c_\Psi(\wedge(\Psi)) = 0$, completing the proof that Meta is NP-hard.

To obtain the second result, we assume both the triangle conjecture and ETH. In this case, we apply the Sparsification Lemma [44] to the initial 3-SAT formula F before invoking the reduction. By the Sparsification Lemma, it is possible in time $2^{o(n)}$ to construct $2^{o(n)}$ formulas $F_1, F_2, \ldots$ such that F is satisfiable if and only if at least one of the F_i is satisfiable. Additionally, each F_i has $O(n)$ clauses. As before, for each such F_i we obtain in polynomial time a corresponding complex Δ_i, whose ground set has size $O(n)$. For each Δ_i, the algorithm from Lemma 48 either outputs its reduced Euler characteristic $\hat{\chi}(\Delta_i)$ or or outputs a UCQ Ψ_i which has the property that that $c_\Psi(\wedge(\Psi_i)) = -\hat{\chi}(\Delta_i)$.

If, for any i, the algorithm outputs a value $\hat{\chi}(\Delta_i) \neq 0$ then F_i is satisfiable, so F is satisfiable. Let I be the set of indices i such that the algorithm outputs a UCQ Ψ_i. The argument from the first result shows that F is satisfiable if and only if there is an $i \in I$ such that counting answers to Ψ_i can be done in linear time.

We will argue that a $2^{o(\ell)}$ algorithm for Meta (where ℓ is the number of CQs in its input) would make it possible to determine in $2^{o(n)}$ time whether there is an $i \in I$ such that counting answers to Ψ_i can be done in linear time. This means that a $2^{o(\ell)}$ algorithm for Meta would make it possible to determine in $2^{o(n)}$ time whether F is satisfiable, contrary to ETH.

To do this, we just need to show that the number of CQs in Ψ_i, which we denote $\ell(\Psi_i)$, is $O(n)$. This follows since the ground set of Δ_i has size $O(n)$ and by Item 4 of Lemma 48, $\ell(\Psi_i)$ is at most the size of the ground set. Since the size of I is $2^{o(n)}$ the running time for determining whether F is satisfiable is $2^{o(n)}$ (for the sparsification) plus $|I|\,\mathrm{poly}(n) = 2^{o(n)}$ time (for computing the complexes Δ_i) plus $\sum_{i \in I} 2^{o(\ell(\Psi_i))} = 2^{o(n)}$ for the calls to Meta, contradicting ETH, as desired. □

The proofs of the following two lemmas are analogous, with the only exception that we do not invoke Theorem 35 but apply Lemma 43 with $d = 1$ to obtain a t such that for each k the function $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{K}_t^k \to \mathcal{D})$ cannot be evaluated in linear time. (For Lemma 50, we also need that SETH implies non-uniform ETH, which is however a standard application of the Sparsification Lemma [43]).

Lemma 50. *If SETH is true then Meta is* NP*-hard and cannot be solved in time $2^{o(\ell)}$. This remains true even if the input to Meta is restricted to be over a binary signature.*

Lemma 51. *If non-uniform ETH is true then Meta is* NP*-hard and, furthermore,*

$$\textsc{Meta} \notin \bigcap_{\varepsilon > 0} \mathrm{DTime}(2^{\varepsilon \cdot \ell})\,.$$

This remains true even if the input to Meta is restricted to be over a binary signature.

Theorem 5 now follows immediately from Lemmas 36, 49, 50, and 51.

Finally, we point out that our construction shows, in fact, something much stronger than just the intractability of deciding whether we can count answers to a UCQ in linear time: For any pair (c, d) of positive integers satisfying $c \leq d$, it is hard to distinguish whether counting answers to a given UCQ can be done in time $O(n^c)$, or whether it takes time at least $\omega(n^d)$. Formally, we introduce the following gap problem:

Definition 52. Let c and d be positive integers with $c \leq d$. The problem $\textsc{Meta}[c, d]$ has as input a union of quantifier-free, self-join-free, and acyclic conjunctive queries Ψ. The goal is to decide whether the function $\mathcal{D} \mapsto \mathsf{ans}(\Psi \to \mathcal{D})$ can be computed in time $O(|\mathcal{D}|^c)$, or whether it cannot be solved in time $O(|\mathcal{D}|^d)$; the behaviour may be undefined for inputs Ψ for which the best exponent in the running time is in the interval $(c, d]$.

Theorem 53. *Assume that non-uniform ETH holds. Then for each positive integer d, the problem $\textsc{Meta}[1, d]$ is* NP*-hard and, furthermore,*

$$\textsc{Meta}[1, d] \notin \bigcap_{\varepsilon > 0} \mathrm{DTime}(2^{\varepsilon \cdot \ell})\,.$$

This remains true even if the input to Meta is restricted to be over a binary signature.

Proof. By Lemma 43 there is a positive integer t such that for all positive integers k, the function $\mathcal{D} \mapsto \mathsf{hom}(\mathcal{K}_t^k \to \mathcal{D})$ cannot be computed in time $O(|\mathcal{D}|^d)$. Fix this t and proceed similarly to the proof of Lemma 51. □

Corollary 54 is an immediate consequence, since any algorithm that solves $\textsc{Meta}[c, d]$ for $1 \leq c \leq d$ solves, without modification, $\textsc{Meta}[1, d]$.

Corollary 54. *Assume that non-uniform ETH holds. Then for every pair (c, d) of positive integers satisfying $c \leq d$, the problem $\textsc{Meta}[c, d]$ is* NP*-hard and, furthermore,*

$$\textsc{Meta}[c, d] \notin \bigcap_{\varepsilon > 0} \mathrm{DTime}(2^{\varepsilon \cdot \ell})\,.$$

This remains true even if the input to Meta is restricted to be over a binary signature.

D CONNECTION TO THE WEISFEILER-LEMAN-DIMENSION

Recall that we call a database a *labelled graph* if its signature has arity at most 2, and if it does not contain a self-loop, that is, a tuple of the form (v, v). Moreover, (U)CQs on labelled graphs must also have signatures of arity at most 2 and must not contain atoms of the form $R(v, v)$, where R is any relation symbol of the signature.

Neuen [55] and Lanzinger and Barceló [49] determined the WL-dimension of computing finite linear combinations of homomorphism counts to be the *hereditary treewidth*, defined momentarily. Since counting homomorphisms is equivalent to counting answers to quantifier-free conjunctive queries, and since the number of answers of a union of conjunctive queries can be expressed as a linear combination of conjunctive query answer counts (Lemma 23), we can state their results as follows.

Definition 55 (Hereditary Treewidth of UCQs)**.** Let Ψ be a UCQ. The *hereditary treewidth* of Ψ, denoted by $\mathsf{hdtw}(\Psi)$, is defined as follows:

$$\mathsf{hdtw}(\Psi) = \max\{\mathsf{tw}(A, X) \mid c_\Psi(A, X) \neq 0\},$$

that is, $\mathsf{hdtw}(\Psi)$ is the maximum treewidth of any conjunctive query that survives with a non-zero coefficient when Ψ is expressed as a linear combination of conjunctive queries.

Then, applying the main result of Neuen, Lanzinger and Barceló to UCQs, we obtain:

Theorem 56 ([49, 55])**.** *Let Ψ be a quantifier-free UCQ on labelled graphs. Then $\mathsf{dim}_{\mathrm{WL}}(\Psi) = \mathsf{hdtw}(\Psi)$.*

This enables us to prove Theorem 7, which we restate for convenience.

Theorem 7. *There is an algorithm that computes a $O(\sqrt{\log k})$-approximation of the WL-dimension k of a quantifier-free UCQ on labelled graphs $\Psi = \varphi_1 \vee \cdots \vee \varphi_\ell$ in time $|\Psi|^{O(1)} \cdot O(2^\ell)$.*

Moreover, let $f : \mathbb{Z}_{>0} \to \mathbb{Z}_{>0}$ be any computable function. The problem of computing an f-approximation of $\mathsf{dim}_{\mathrm{WL}}(\Psi)$ given an input UCQ $\Psi = \varphi_1 \vee \cdots \vee \varphi_\ell$ is NP-hard, and, assuming ETH, an f-approximation of $\mathsf{dim}_{\mathrm{WL}}(\Psi)$ cannot be computed in time $2^{o(\ell)}$.

Proof. For the upper bound, we compute the coefficients

$$c_\Psi(\mathcal{A}, X) = \sum_{\substack{J \subseteq [\ell] \\ \wedge(\Psi|_J) \cong (\mathcal{A}, X)}} (-1)^{|J|+1},$$

that is, for each subset $J \subseteq [\ell]$, compute $\wedge\left(\Psi|_J\right)$. Afterwards, collect the isomorphic terms and compute $c_\Psi(\wedge\left(\Psi|_J\right))$ for each $J \subseteq [\ell]$. Clearly, $c_\Psi(\mathcal{A}, X) = 0$ for every $(\mathcal{A}, X)$ that is not isomorphic to any $\wedge\left(\Psi|_J\right)$. Clearly, this can be done in time $|\Psi|^{O(1)} \cdot O(2^\ell)$.

Next, for each $(\mathcal{A}, X)$ with $c_\Psi(\mathcal{A}, X) \neq 0$, we use the algorithm of Feige, Hajiaghayi, and Lee [33] to compute in polynomial time a g-approximation $S(\mathcal{A}, X)$ of the treewidth of $(\mathcal{A}, X)$, where $g(k) \in O(\sqrt{\log k})$. Finally, we output the maximum of $S(\mathcal{A}, X)$ over all $(\mathcal{A}, X)$ with $c_\Psi(\mathcal{A}, X) \neq 0$

For the lower bound, assume that there is a function $f : \mathbb{Z}_{>0} \to \mathbb{Z}_{>0}$ and an algorithm $\mathbb{A}$ that computes an f-approximation of $\mathsf{dim}_{\mathrm{WL}}(\Psi)$ in subexponential time in the number of conjunctive queries in the union. We will use $\mathbb{A}$ to construct a subexponential time algorithm for 3-SAT, which refutes ETH. Our construction is similar to the proof of Lemma 49. Fix any positive integer $t > f(1) + 1$.

Let F be a 3-CNF with n variables, which we can again assume to be sparse by using the Sparsification Lemma [19] (the details are identical to its application in the proof of Lemma 49). Next, using [58], we obtain a complex Δ, the reduced Euler characteristic of which is zero if and only if F is not satisfiable. Finally, we apply Lemma 48 with our choice of t. The corresponding algorithm computes in polynomial time either the reduced Euler characteristic of Δ, or otherwise outputs a UCQ Ψ such that the number ℓ of CQs in the union is bounded by $O(n)$. Moreover, the hereditary treewidth of Ψ is 1 if $\hat{\chi}(\Delta) = 0$, i.e., if F is not satisfiable; and its hereditary treewidth is at least $\mathsf{tw}(\mathcal{K}_t^k) = t - 1 > f(1)$, otherwise.

Thus, we run $\mathbb{A}$ on Ψ and report that F is satisfiable if and only if it outputs $S > f(1)$. Since $\mathbb{A}$ runs in time $2^{o(\ell)}$, the total running time is bounded by $2^{o(n)}$, which refutes ETH. NP-hardness follows likewise. □

Finally, the proof of Theorem 8 is identical with the only exception being that, since k is fixed, we can substitute the approximation algorithm for treewidth of Feige, Hajiaghayi, and Lee [33] by the exact algorithm of Bodlaender [16], which runs in polynomial time if k is fixed.

E NECESSITY OF THE SIDE CONDITIONS IN THEOREM 3

We show that Theorem 3 is optimal in the sense that, if any of the conditions (I), (II), or (III) is dropped, the statement of the theorem becomes false (assuming that W[1]-hard problems are not fixed-parameter tractable).

Dropping condition (I).

Lemma 57. *There is a recursively enumerable class C of quantifier-free UCQs of bounded arity such that $\wedge(C)$ has unbounded treewidth but #UCQ(C) is fixed-parameter tractable. The class C satisfies (II) and (III).*

Proof. Let Δ be the second complex in Figure 1, that is, the ground set is $\Omega = \{1, 2, 3, 4\}$ and the facets are $\{1, 2\}, \{2, 3\}, \{3, 1\}$, and $\{4\}$. Note that Δ is irreducible (no element dominates another element), non-trivial, and Ω is not a facet. We can thus use Lemma 46 and let Ψ_t to be the output of algorithm $\hat{\mathbb{A}}_t$ given Δ. Note that Ψ_t is quantifier-free — in particular, this implies that all conjunctive queries within Ψ_t are #minimal. Since the algorithm $\hat{\mathbb{A}}_t$ can be explicitly constructed from t (see Lemma 46 and Algorithm 1) the class $C = \{\Psi_t \mid t \geq 1\}$ is recursively enumerable. Furthermore, all of the relation symbols in queries in UCQs in C have arity 2, so C has bounded arity.

By Item 1 of Lemma 46, $\wedge(C)$ has unbounded treewidth, since the treewidth of $\mathcal{K}_t^k$ is equal to $t-1$. Moreover, by Item 5 of Lemma 46, all Ψ_t are unions of self-join-free conjunctive queries. We next show that #UCQ(C) is fixed-parameter tractable.

Recall that $\hat{\chi}(\Delta) = -(3-4+1) = 0$. Item 2 of Lemma 46 shows that $c_{\Psi_t}(\wedge(\Psi_t)) = 0$. Item 3 shows that for any relational structure $\mathcal{B}$ that is not isomorphic to $\wedge(\Psi_t)$ with $c_{\Psi_t}(\mathcal{B}) \neq 0$, $\mathcal{B}$ is acyclic. Recall that $\Gamma(C)$ is the class of those conjunctive queries that contribute to the CQ expansion of at least one UCQ in C. So all CQs in $\Gamma(C)$ are acyclic, which means that the treewidth of $\Gamma(C)$ is bounded by 1.

Since each query in $\Gamma(C)$ is quantifier-free, and is thus its own contract, $\Gamma(C) = \mathsf{contract}(\Gamma(C))$. Thus, by Theorem 1, #UCQ(C) is fixed-parameter tractable.

We finish the proof by showing that C satisfies (II) and (III). Item (II) - that the number of existentially qunatified variables of queries in C is bounded - is trivial, because there are none. We have already noted (III) – that the UCQs in C are unions of self-join-free CQs. ◻

Dropping condition (II).

Lemma 58. *There is a recursively enumerable and deletion-closed class C of unions of self-join-free conjunctive queries of bounded arity such that $\wedge(C)$ has unbounded treewidth but #UCQ(C) is fixed-parameter tractable.*

Proof. The statement of the lemma guarantees that C satisfies items (I) and (III) of Theorem 3.

Let $k \geq 3$ be a positive integer and let $\tau_k = (E_1, \dots, E_k)$ be a signature with $\mathsf{arity}(E_i) = 2$ for all $i \in [k]$. For any pair $i, j \in [k]$ with $i < j$, consider the conjunctive query

$$\varphi_k^{i,j}(x_1, \dots, x_k, x_\perp) = \exists y_k^{i,j} : E_i(x_i, y_k^{i,j}) \wedge E_j(x_j, y_k^{i,j}) \wedge \bigwedge_{\ell \in [k] \setminus \{i,j\}} E_\ell(x_\ell, x_\perp).$$

Let $\Psi_k = \bigvee_{i<j \in [k]} \varphi_k^{i,j}$ and let C be obtained from the class $\{\Psi_k \mid k \geq 3\}$ by taking the closure under deletion of conjunctive queries. Clearly, C is recursively enumerable.

Note that each conjunctive query $\varphi_k^{i,j}$ is self-join-free and that Ψ_k contains $\binom{k}{2}$ existentially quantified variables. Thus the number of existentially quantified variables of queries in C is unbounded. Moreover, the treewidth of $\wedge(C)$ is unbounded. To see this, observe that

$$\wedge(\Psi_k)(x_1, \dots, x_k) = \exists y_k^{1,2}, y_k^{1,3}, \dots, y_k^{k-1,k} : \bigwedge_{i<j} E_i(x_i, y_k^{i,j}) \wedge E_j(x_j, y_k^{i,j}) \wedge \bigwedge_{\ell \in [k]} E_\ell(x_\ell, x_\perp).$$

Therefore, the Gaifman graph of $\wedge(\Psi_k)$ contains as a subgraph a subdivision of a k-clique and thus has treewidth at least $k-1$.

It remains to show that #UCQ(C) is fixed-parameter tractable. To show this we claim that the classes $\Gamma(C)$ and $\mathsf{contract}(\Gamma(C))$ have treewidth at most 2, and thus the problem #UCQ(C) is fixed-parameter tractable by Theorem 1. To prove the claim, fix any $k \geq 3$ and any non-emtpy subset $J \subseteq \{(i, j) \in [k^2] \mid i < j\}$. We will show that the #minimal representatives of $\wedge(\Psi_k, J)$ and its contract are acyclic. To this end, assume first that $|J| = 1$. Then $\wedge(\Psi_k, J)$ is equal to one of the conjunctive queries $\varphi_k^{i,j}$, which is clearly acyclic. Since $\varphi_k^{i,j}$ is self-join-free and does not contain isolated variables, it is #minimal by Lemma 30. Let G be the Gaifman graph of $\varphi_k^{i,j}$, and recall that the contract of $\varphi_k^{i,j}$ is obtained from $G[X]$ by adding an edge between two free variables in X if and only if there is a connected component in the quantified variables that is adjacent to both free variables. Since the only quantified variable in $\varphi_k^{i,j}$ is $y_k^{i,j}$, which is adjacent (in G) to x_i and x_j, the contract of $\varphi_k^{i,j}$ is just the graph obtained from $G[X]$ by adding an edge between x_i and x_j, which also yields an acyclic graph.

Next assume that $|J| \geq 2$. For an index $s \in [k]$, we say that J *covers* s if each $(i, j) \in J$ satisfies $i = s$ or $j = s$. We distinguish three cases:

(A) There are distinct $s_1 < s_2$ such that J covers s_1 and s_2. Then $J = \{(s_1, s_2)\}$, contradicting the assumption that $|J| \geq 2$.

(B) There is precisely one $s \in [k]$ such that J covers s. Assume w.l.o.g. that $s = k$. Then since every $(i, j) \in J$ has $i < j$

$$\begin{aligned}\wedge(\Psi_k, J) &= \bigwedge_{\ell \in [k-1]} E_\ell(x_\ell, x_\perp) \wedge \bigwedge_{(i,j) \in J} \exists y_k^{i,j} : E_i(x_i, y_k^{i,j}) \wedge E_j(x_j, y_k^{i,j}) \\ &= \bigwedge_{\ell \in [k-1]} E_\ell(x_\ell, x_\perp) \wedge \bigwedge_{(i,k) \in J} \exists y_k^{i,k} : E_i(x_i, y_k^{i,k}) \wedge E_k(x_k, y_k^{i,k}).\end{aligned}$$

Observe that any answer of $\bigwedge(\Psi_k, J)$ in a database $\mathcal{D}$ is also an answer of the following query, and vice versa:

$$\psi_k := \bigwedge_{\ell \in [k-1]} E_\ell(x_\ell, x_\perp) \wedge \bigwedge_{i \in [k-1]} \exists y_k^{i,k} : E_i(x_i, y_k^{i,k}) \wedge E_k(x_k, y_k^{i,k}),$$

since all $y_k^{i,\ell}$ with $i < k$ can be mapped to the same vertex as $x_\perp$. Thus $\bigwedge(\Psi_k, J)$ and ψ_k are counting equivalent. Moreover, ψ_k is self-join-free and does not contain isolated variables. Thus, by Lemma 30, it is #minimal. Finally, deleting x_k from the Gaifman graph of ψ_k yields an acyclic graph, and the same is true for the contract of ψ_k. Therefore, the treewidth of both ψ_k and its contract are at most 2.

(C) There is no $s \in [k]$ such that J covers s. Then

$$\bigwedge(\Psi_k, J) = \bigwedge_{\ell \in [k]} E_\ell(x_\ell, x_\perp) \wedge \bigwedge_{(i,j) \in J} \exists y_k^{i,j} : E_i(x_i, y_k^{i,j}) \wedge E_j(x_j, y_k^{i,j}).$$

Observe that any answer of $\bigwedge(\Psi_k, J)$ in a database $\mathcal{D}$ is also an answer of the following query, and vice versa:

$$\psi_k := \bigwedge_{\ell \in [k]} E(x_\ell, x_\perp),$$

since all $y_k^{i,j}$ can be mapped to the same vertex as $x_\perp$.
Thus $\bigwedge(\Psi_k, J)$ and ψ_k are counting equivalent. Since ψ_k does not contain quantified variables, it must be both its own #core and its own contract. This concludes the proof of the claim since ψ_k is acyclic.

□

Dropping condition (III).

Lemma 59. *There is a recursively enumerable and deletion-closed class C of quantifier-free UCQs of bounded arity such that $\bigwedge(C)$ has unbounded treewidth but #UCQ(C) is fixed-parameter tractable.*

Proof. The statement of the lemma guarantees that C satisfies items (I) and (II) of Theorem 3. We show an even stronger claim by requiring C to be a recursively enumerable class of quantifier-free CQs (instead of UCQs) of bounded arity such that $\bigwedge(C)$ has unbounded treewidth but #UCQ(C) is polynomial-time solvable. Note that each conjunctive query is a (trivial) union of conjunctive queries; moreover, this also means that C is deletion-closed.

For each $k \geq 1$, we define a conjunctive query ψ_k over the signature of graphs as follows:

$$\psi_k(x_1, \dots, x_k, x_\perp) = \exists y : \bigwedge_{i \in [k]} E(x_i, x_\perp) \wedge E(x_i, y).$$

The query ψ_k has only one quantified variable. Moreover, the contract of ψ_k is a k-clique and thus has treewidth $k-1$. However, ψ_k is clearly #equivalent to the query

$$\psi'_k = \bigwedge_{i \in [k]} E(x_i, x_\perp),$$

which is its own contract (since there are no quantified variables), and which is of treewidth 1. Thus, for C being the class of all ψ_k, we find that contract($\bigwedge(C)$) has unbounded treewidth, but, according to Theorem 18, the problem #UCQ(C) is solvable in polynomial time. □