Efficient Evaluation and Approximation of Well-designed Pattern Trees

Pablo Barceló
Center for Semantic Web Research & Dept. of Comp. Science, University of Chile
pbarcelo@dcc.uchile.cl

Reinhard Pichler
Faculty of Informatics
Vienna University of Technology
pichler@dbai.tuwien.ac.at

Sebastian Skritek
Faculty of Informatics
Vienna University of Technology
skritek@dbai.tuwien.ac.at

ABSTRACT

Conjunctive queries (CQs) fail to provide an answer when the pattern described by the query does not exactly match the data. CQs might thus be too restrictive as a querying mechanism when data is semistructured or incomplete. The semantic web therefore provides a formalism – known as *well-designed pattern trees* (WDPTs) – that tackles this problem: WDPTs allow us to match patterns over the data, if available, but do not fail to give an answer otherwise. Here we abstract away the specifics of semantic web applications and study WDPTs over arbitrary relational schemas.

Our language properly subsumes the class of CQs. Hence, WDPT evaluation is intractable. We identify structural properties of WDPTs that lead to tractability of various variants of the evaluation problem. For checking if a WDPT is equivalent to one in our tractable class, we prove 2EXPTIME-membership. As a corollary, we obtain fixed-parameter tractability of (variants of) the evaluation problem. Our techniques also allow us to develop a theory of approximations for WDPTs.

Categories and Subject Descriptors

H.2.3 [**Database Management**]: Languages—*Query Languages*

General Terms

Algorithms, Theory

Keywords

RDF; well-designed pattern trees; efficient query answering; containment; subsumption; approximations

1. INTRODUCTION

Conjunctive queries (CQs) constitute the core of the query languages for relational databases and also the most intensively studied querying mechanism in the database theory community. But CQs suffer from a serious drawback when dealing with information that is semistructured or incomplete, or when users do not have a good understanding of the schema that underlies the data: CQs fail to provide an answer when the pattern described by the query does not exactly match the data.

The semantic web therefore provides formalisms to overcome this problem [20]. We concentrate on the simplest such formalism, which corresponds to the {AND,OPT}-fragment of SPARQL – the standard query language for the semantic web data model (RDF). This fragment allows the user not only to specify patterns by taking conjunctions of atoms (using the AND-operator) – in the same way as CQs do – but also to match patterns over the data, if available, without failing to give an answer otherwise. This is precisely the role of the OPT-operator, which allows for optional matching and essentially corresponds to the left-outer join operator in the relational algebra.

Example 1. Consider the following {AND,OPT}-SPARQL query that is posed over a database that stores information about bands and records:[1]

$$\Big(\big((x, \texttt{recorded\_by}, y)\ \mathsf{AND}\ (x, \texttt{published}, \text{“after\_2010”})\big)\ \mathsf{OPT}\ (x, \texttt{NME\_rating}, z)\Big)\ \mathsf{OPT}\ (y, \texttt{formed\_in}, z'). \quad (1)$$

This query retrieves all pairs (b, r) such that r is a record of band b that was published during this decade. This is specified by the pattern $(x, \texttt{recorded\_by}, y)$ AND $(x, \texttt{published\_in}, \text{“after\_2010”})$. Furthermore, whenever possible this query also retrieves (one or both of) the following pieces of data: the rating t of record r as declared by the NME magazine and the year t' in which band b was formed. In other words, in addition to (b, r) we also retrieve t and/or t' if they can be found in the database. This is specified by the atoms $(x, \texttt{NME\_rating}, z)$ and $(y, \texttt{formed\_in}, z')$ following the respective OPT-operators.

Pérez et al. noticed that a non-constrained interaction of the operators AND and OPT in SPARQL may lead to undesired behavior [18]. This motivated the definition of a better behaved syntactic restriction of the language, known as *well-designed* {AND,OPT}-SPARQL. In particular, the query in Example 1 is well-designed. Among other things,

[1]Note that we are using here the more algebraic-style notation of [18] rather than the official SPARQL syntax of [20].

PODS'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.

http://dx.doi.org/10.1145/2745754.2745767 .

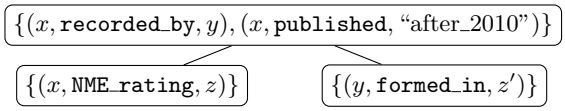

Figure 1: WDPT representing query (1) from Example 1.

queries in this fragment are more natural than the full language [18], allow for lower complexity of evaluation [18], and lend themselves to optimization techniques [17, 19]. Moreover, they allow for a natural tree representation, known as *well-designed pattern trees*, or WDPTs [17].

Intuitively, a WDPT p consists of a tree T rooted in a distinguished node r and a function that labels each node of T with a set of RDF atoms. The condition of being well-designed imposes that appearances of the same variable in different nodes of T must be connected. Each node of a WDPT p represents a conjunction of atoms, while the nesting of optional matching is represented by the tree structure of p. For instance, the query in Example 1 can be represented as the WDPT in Figure 1.

The semantics of WDPT p is as follows. With each subtree T' of T rooted in r we associate a CQ $r_{T'}$ defined by the conjunction of all atoms in the nodes of T'. The evaluation of WDPT p over database $\mathcal{D}$ consists then of all "maximal" answers to the CQs of the form $r_{T'}$. That is, we take the union of all answers to the CQs of the form $r_{T'}$, for T' a subtree of T rooted in r, and then remove all those answers that are "extended" by some other answer in the set. We revisit Example 1 to illustrate these ideas.

Example 2. Consider an RDF database $\mathcal{D}$ consisting of triples ("Our_love", `recorded_by`, "Caribou"), ("Our_love", `published`, "after_2010"), ("Swim", `recorded_by`, "Caribou"), ("Swim", `published`, "after_2010"), ("Swim", `NME_ranking`, "2"). The evaluation over $\mathcal{D}$ of the WDPT in Figure 1, and, therefore, of the query in (1), consists of partial mappings μ_1 and μ_2 defined on variables x, y, z, z' such that: (1) μ_1 is only defined on x and y in such a way that $\mu_1(x)$ = "Our_love" and $\mu_2(y)$ = "Caribou", and (2) μ_2 is defined on x, y and z in such a way that $\mu_1(x)$ = "Swim", $\mu_2(y)$ = "Caribou", and $\mu_2(z)$ = "2".

The expressive power of WDPTs is limited due to the absence of projection, a feature that CQs enjoy. Consequently, WDPTs are often enhanced with projection as a way to increase their expressiveness and to obtain a proper extension of the class of CQs over RDF vocabularies [17, 19]. In this paper we concentrate on this extended class of WDPTs.

Example 3. For the WDPT from Example 1, one might decide to project out the variable x. This would result in restricting the mappings μ_1 and μ_2 from Example 2 to μ_1' and μ_2' in such a way that: (1) μ_1' is only defined on y with $\mu_1'(y)$ = "Caribou", and (2) μ_2' is defined on y and z and it holds that $\mu_2'(y)$ = "Caribou" and $\mu_2'(z)$ = "2".

Our view is that WDPTs are of interest not only for semantic web applications, but also for every application that needs to handle semistructured or incomplete data. This motivates our study of WDPTs over arbitrary relational schemas, abstracting away from the specifics of the semantic web data model – RDF – which only allows for triples in the nodes of WDPTs (or, in other words, relational atoms over a single ternary relation).

Despite the importance of WDPTs, very little is known about some fundamental problems related to them. In particular, no in-depth study has been carried out regarding efficient evaluation of these queries, a problem that permeates the literature on CQs and its extensions [21, 12, 13, 16, 5]. Likewise, restrictions on WDPTs to decrease the complexity of basic static query analysis tasks [19] such as testing containment are largely unexplored. Topics strongly related to the identification of tractable fragments of query evaluation are semantic query optimization and query approximation. There we ask if some query is equivalent to or can at least be "approximated" by a query from a tractable class. These questions have recently received quite some interest in case of CQs and conjunctive regular path queries over graph databases [10, 5, 4]. So far, nothing is known in case of WDPTs.

The main goal of this work is to initiate a systematic study of tractable fragments of WDPTs for query evaluation and to apply these fragments to fundamental questions in the areas of query analysis, semantic optimization, and approximation.

Efficient evaluation of WDPTs. Evaluation of WDPTs is defined in terms of CQ evaluation, which is an intractable problem in general. Therefore, our goal of identifying tractable classes of WDPTs naturally calls for a restriction of the classes of CQ patterns allowed in them. In particular, there has been a flurry of activity around the topic of determining which classes of CQs admit efficient evaluation that could be reused in our scenario [21, 12, 13]. We concentrate here on two of the most fundamental classes: those of bounded *treewidth* [8, 10] and *hypertreewidth* [13]. We denote by $\mathsf{TW}(k)$ and $\mathsf{HW}(k)$ the CQs of treewidth and hypertreewidth at most k, for $k \geq 1$. Queries in these classes even lie in the parallelizable class LogCFL [12, 13].

The restriction to tractable classes of CQ-evaluation has already been successfully applied in the context of WDPTs without projection. It is known, in particular, that a very mild condition known as *local tractability* leads to efficient evaluation [17]. This condition enforces each node in the WDPT to contain a set of relational atoms from one of our tractable classes of CQs, namely $\mathsf{TW}(k)$ or $\mathsf{HW}(k)$. Nevertheless, it is also known that this condition does not lead to tractability for the more expressive WDPTs with projection that we study here [17]. Then the question remains: When is the evaluation of WDPTs tractable or, more precisely, which natural conditions can be added to local tractability to achieve tractable WDPT evaluation? We shall identify such a condition – called *bounded interface* – that limits by a constant the number of variables that each node in a WDPT can share with its children. Notably, similar conditions have been recently applied to obtain reasonable bounds for the containment problem of Datalog into unions of CQs [6]).

Due to the nature of WDPTs, two other evaluation problems – called the *partial* and *maximal* evaluation problems – are of importance [18, 2]. The first one refers to checking whether a mapping μ is a *partial answer* to the evaluation $p(\mathcal{D})$ of a WDPT p over a database $\mathcal{D}$; i.e., whether there is a mapping $\mu' \in p(\mathcal{D})$ that "extends" μ. The second problem asks if μ is maximal among all answers in $p(\mathcal{D})$. (In the pres-

ence of projection, it may happen that some partial mapping and also a proper extension of this mapping are solutions of a WDPT. E.g., in Example 3, μ_1' and also its extension μ_2' are solutions.) We shall identify tractable fragments also for these problems by introducing the notion of *global tractability*. Here, we restrict every CQ $r_{T'}$ represented by a subtree T' of the WDPT T to belong to $\mathsf{TW}(k)$ or $\mathsf{HW}(k)$. We show that global tractability suffices to ensure tractability of the latter problems even though it is a weaker condition than local tractability plus bounded interface.

Containment and subsumption. Containment is a crucial static analysis task that amounts to checking whether the evaluation of a query q is necessarily contained in the evaluation of another query q' (often written as $q \subseteq q'$). The containment problem for CQs is known to be NP-complete [7]. In contrast, it becomes undecidable for WDPTs [19] and remains so even for our restriction to local tractability and bounded interface. The same holds for the equivalence problem (i.e., checking whether the evaluation of q necessarily coincides with the evaluation of q').

It is known that WDPT containment may display some unintuitive behavior, which motivated the introduction of a meaningful variant of it known as *subsumption* [3]. This is the problem of checking whether every answer of a WDPT p over any database $\mathcal{D}$ can be "extended" to an answer of WDPT p' over $\mathcal{D}$ (we denote this by $p \sqsubseteq p'$). The corresponding notion of equivalence is then *subsumption-equivalence*, where we ask if both directions $p \sqsubseteq p'$ and $p' \sqsubseteq p$ hold. In sharp contrast to containment, subsumption for WDPTs is known to be decidable and complete for the class Π_2^P [17]. Subsumption-equivalence can be shown to have the same behavior. We will investigate in this context whether restrictions to tractable classes of WDPT evaluation alleviate the complexity of checking subsumption or subsumption-equivalence. Our main result will be that the restriction to tractable classes of query evaluation allows us to reduce the complexity to coNP but not any further.

Semantic optimization of WDPTs. We introduce several syntactic restrictions on WDPTs that lead to tractability of (variants of) evaluation. As a general method for finding larger classes of queries with good evaluation properties, one typically explores the *semantic space* defined by the syntactical restrictions that yield tractability; this space is defined by all queries that are equivalent to one in the well-behaved class (see, e.g., [10, 4, 5]). In this context the following are the two most important questions:

1. Is it decidable to check whether a query is equivalent to one in the well-behaved syntactically defined class?
2. Can the evaluation problem be solved more efficiently for queries equivalent to one in a well-behaved class?

Positive answers to these questions have been provided in the context of CQs [10] and conjunctive regular path queries over graph databases [5]. For example, regarding question (1) it is known that verifying if a CQ is equivalent to one in $\mathsf{TW}(k)$ is in NP. For question (2) it can be proved that the evaluation problem for those CQs that are equivalent to one in $\mathsf{TW}(k)$ is in Ptime [10]. Here we investigate these questions for WDPTs.

Some care is required in fixing the appropriate setting for this investigation. For instance, since classical equivalence is undecidable for WDPTs we have to content ourselves with the relaxed notion of equivalence based on subsumption introduced earlier. But subsumption-equivalence only preserves partial and maximal answers. We shall therefore focus on the partial and maximal evaluation problems and choose global tractability as the corresponding tractability criterion of WDPTs. Our main finding will be a positive answer to both questions (1) and (2) above in this setting.

Approximations of WDPTs. When a query q is not equivalent to one in a well-behaved class $\mathcal{Q}$ it might be convenient to compute a $\mathcal{Q}$-*approximation* of q. This is a query $q' \in \mathcal{Q}$ that is maximal (with respect to $\subseteq$) among all queries in $\mathcal{Q}$ that are contained in q. Intuitively, q' is sound with respect to q (since $q' \subseteq q$) and provides the best approximation to q among all queries in $\mathcal{Q}$ that are sound for $\mathcal{Q}$.

The notion of approximations is by now well-understood in the context of CQs [4]. For instance, $\mathsf{TW}(k)$-approximations of CQs always exist and can be computed in single-exponential time. These results allow us to explain the role of approximations. In general, the evaluation of a CQ q on a database $\mathcal{D}$ is of the order $|\mathcal{D}|^{O(|q|)}$, which might be prohibitively expensive for a large dataset $\mathcal{D}$ even if q is small. On the other hand, the previous properties imply that computing and running an approximation of a CQ q on a database $\mathcal{D}$ takes time $O(|\mathcal{D}| \cdot 2^{t(|q|)})$, for some polynomial $t : \mathbb{N} \to \mathbb{N}$. This is much faster than $|\mathcal{D}|^{O(|q|)}$ on large databases. Thus, if the quality of the approximation is good, we may prefer to run this faster query instead of q.

Our techniques allow us to develop a thorough theory of approximations for WDPTs. Again, we define approximations via subsumption instead of containment. Furthermore, we look for approximations by WDPTs of the globally tractable classes. Our main finding is that approximations in these classes always exist, can be computed in double-exponential time, and have at most single-exponential size.

Unions of WDPTs. We finally study unions of WDPTs (UWDPTs) as a natural extension of WDPTs. For the variants of query evaluation considered here, all results on WDPTs easily carry over to UWDPTs. In contrast, for semantic optimization and approximation by tractable classes of UWDPTs, we shall reveal a huge difference between WDPTs and UWDPTs. By establishing a close connection between UWDPTs and unions of CQs, we can apply the theory of approximations of CQs to WDPTs. This will allow us to prove significantly better complexity bounds for the problems studied in the context of semantic optimization and approximation.

Organization and main results. In Section 2, we recall some basic notions and results on CQs and WDPTs. A conclusion and outlook to future work are given in Section 7. Our main results are detailed in Sections 3 – 6, namely:

• The problem of finding tractable classes of WDPTs is studied in Section 3. Our main result states that WDPTs which enjoy local tractability and bounded interface can be evaluated in LogCFL. Since our classes properly contain CQs of bounded treewidth and hypertreewidth, we obtain relevant extensions of these well-known tractable classes of CQs. We also study two further variants of query evaluation, namely partial evaluation and maximal evaluation. We show that global tractability suffices to ensure tractability of these two problems. Interestingly, it does not suffice to obtain tractability for the exact evaluation problem. In fact, we

show that global tractability is a strictly weaker condition than local tractability plus bounded interface.

• We dedicate Section 4 to the study of containment and subsumption. Subsumption has already been known to be Π_2^P-complete [17]. We establish the same complexity classification for subsumption-equivalence and show that Π_2^P-completeness of both subsumption and subsumption-equivalence continues to hold even under the restriction to local tractability. This complexity is then shown to drop to coNP-completeness under the further restriction of the WDPTs to global tractability. In case of testing subsumption $p \sqsubseteq p'$, we also identify a significant asymmetry in that coNP-membership only depends on the restriction of p', while p may be an arbitrary WDPT.

• Section 5 contains our investigation of semantic optimization and approximations. Our main finding in terms of semantic optimization is that the problem of checking whether a WDPT is subsumption-equivalent to one in our tractable classes is decidable in $\text{NEXPTIME}^{\text{NP}}$. From this we get as immediate corollary that the partial and maximal evaluation problems for those WDPTs which are subsumption-equivalent to one in a tractable classes are *fixed-parameter tractable* (taking the size of the WDPT as parameter).[2] As far as the approximation in the globally tractable classes of WDPTs is concerned, we show that such approximations always exist, can be computed in double-exponential time, and have at most single-exponential size. We also prove that the exponential blowup in the size of a WDPT approximation cannot be avoided. The total time for computing and running an approximation of a WDPT p on a database $\mathcal{D}$ thus takes time $O(|\mathcal{D}| \cdot 2^{2^{t(|p|)}})$, for some polynomial $t : \mathbb{N} \to \mathbb{N}$. For big $\mathcal{D}$, this is in general faster than directly evaluating p over $\mathcal{D}$.

• In Section 6 we study the extension of WDPTs to unions of WDPTs. For the variants of query evaluation considered here, all results on WDPTs easily carry over to UWDPTs. We can thus establish for UWDPTs the analogous tractability results as for WDPTs. For instance, unions of WDPTs that are locally tractable and have bounded interface can be evaluated in LogCFL. The same holds for unions of globally tractable WDPTs in the context of partial and maximal evaluation. In contrast, for semantic optimization and approximation by tractable classes of UWDPTs, we shall reveal a huge difference between WDPTs and UWDPTs. More precisely, we obtain better bounds for the complexity of (1) checking whether a UWDPT ϕ is equivalent to a union ϕ' of tractable WDPTs, and (2) checking if a union ϕ' of tractable WDPTs is an approximation of a UWDPT ϕ. In fact, both problems are Π_2^P-hard. Depending on whether we define global tractability via treewidth or hypertreewidth, we get an upper bound of Π_2^P or Π_3^P, respectively. This is in stark contrast to single WDPTs, where we essentially obtained double exponential upper bounds for the analogous problems.

Proof sketches for most results are provided in Appendix A. Full proofs will be provided in a full version.

[2]Recall that the evaluation problem for a class $\mathcal{Q}$ of queries is fixed-parameter tractable w.r.t. the size of the query, if there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ and a constant $k \geq 1$ such that evaluating a query $q \in \mathcal{Q}$ over a database $\mathcal{D}$ can be done in time $O(|\mathcal{D}|^k \cdot f(|q|))$.

2. PRELIMINARIES

Conjunctive queries. Let $\mathbf{U}$ and $\mathbf{X}$ be disjoint countably infinite sets of constants and variables, respectively. Assume that σ is a relational schema. A *relational atom* over σ is an expression of the form $R(\bar{v})$, where R is a relation symbol in σ of arity $n > 0$ and $\bar{v}$ is an n-tuple over $\mathbf{X} \cup \mathbf{U}$. A *database* $\mathcal{D}$ over σ is a set of relational atoms without variables over σ.

A *conjunctive query* (CQ) q over σ is a rule of the form:

$$\text{Ans}(\bar{x}) \leftarrow R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m), \tag{2}$$

where each $R_i(\bar{v}_i)$ $(1 \leq i \leq m)$ is a relational atom in σ and $\bar{x}$ is a tuple of distinct variables among the ones that appear in the $\bar{v}_i$'s. We often write this CQ as $q(\bar{x})$ in order to denote that $\bar{x}$ is the tuple of *free* variables of q.

The semantics of CQs is defined in terms of *homomorphisms*. Let $\mathcal{D}$ be a database over σ. A homomorphism from a CQ $q(\bar{x})$ of the form (2) to $\mathcal{D}$ is a partial mapping $h : \mathbf{X} \to \mathbf{U}$ such that $R_i(h(\bar{v}_i)) \in \mathcal{D}$,[3] for $1 \leq i \leq m$. We denote by $h_{\bar{x}}$ the restriction of h to the variables in $\bar{x}$. The *evaluation* $q(\mathcal{D})$ of $q(\bar{x})$ over $\mathcal{D}$ is the set of all mappings of the form $h_{\bar{x}}$, such that h is a homomorphism from q to $\mathcal{D}$.[4]

For comparing partial mappings, the notion of subsumption is useful: let $h, h' : \mathbf{X} \to \mathbf{U}$ be partial mappings, and assume that X and X' are the subsets of $\mathbf{X}$ where h and h' are defined, respectively. Then we say that h is *subsumed* by h', denoted $h \sqsubseteq h'$, if $X \subseteq X'$ and $h(x) = h'(x)$, for each $x \in X \cap X'$. If $h \sqsubseteq h'$ but it is not the case that $h' \sqsubseteq h$, then we write $h \sqsubset h'$.

Pattern trees. When data is inherently incomplete, it is convenient to work with a proper extension of the class of CQs known as *pattern trees*. Intuitively, a pattern tree allows the user to specify patterns over the data that should be recovered, if available, but do not force the query to fail to give an answer otherwise.

We concentrate here on the class of *well-designed* pattern trees (WDPTs), which has received considerable attention in the semantic web literature. As shown in [17], WDPTs provide an intuitive representation of well-designed {AND,OPT}-SPARQL [18]. WDPTs have been used extensively in analyzing query evaluation and in static query analysis of SPARQL [17, 18, 19].

Intuitively, the nodes of a WDPT represent CQs (called "basic graph patterns" in the semantic web context) while the tree structure of a WDPT represents the nesting of optional matching. We formalize the class of WDPTs below.

Definition 1. (WDPTs) A *well-designed pattern tree* (WDPT) over a relational schema σ is a tuple $(T, \lambda, \bar{x})$, such that the following holds:

1. T is a tree rooted in a distinguished node r and λ maps each node t in T to a set of relational atoms over σ.
2. For every variable y that appears in T, the set of nodes of T where y is mentioned is connected.
3. We have that $\bar{x}$ is a tuple of distinct variables mentioned in T, which correspond to the *free variables* of the WDPT.

[3]As usual, we write $h(v_1, \ldots, v_n)$ for $(h(v_1), \ldots, h(v_n))$, and define $h(u) = u$ for each constant $u \in \mathbf{U}$.

[4]Our definition of $q(\mathcal{D})$ slightly departs from the traditional one in which $q(\mathcal{D})$ is the set of all tuples of the form $h(\bar{x})$, for h a homomorphism from q to $\mathcal{D}$.

We say that $(T, \lambda, \bar{x})$ is *projection-free*, if $\bar{x}$ contains all variables mentioned in T.

Pairs (T, λ) that satisfy condition (1) correspond to the natural extension of pattern trees studied in the semantic web context to arbitrary schemas. Condition (2) is the one that defines well-designedness [18]. Projection-free WDPTs are of importance in the semantic web context [17, 18].

Assume $p = (T, \lambda, \bar{x})$ is a WDPT over σ. We write r to denote the root of T. Given a subtree T' of T rooted in r, we define $q_{T'}$ to be the CQ $\text{Ans}(\bar{y}) \leftarrow R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)$, where the $R_i(\bar{v}_i)$'s are the relational atoms that label the nodes of T', i.e.,

$$\{R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)\} \;=\; \bigcup_{t \in T'} \lambda(t),$$

and $\bar{y}$ are all the variables that are mentioned in T'.

We write $|p|$ to denote the *size* of p in standard relational notation – which corresponds to the size of CQ q_T. By slight abuse of notation, we often identify nodes and subtrees in p with their labels. For example, we shall speak of a homomorphism from subtree T' to subtree T'' to refer to a mapping between the atoms occurring in the labels of the nodes in these subtrees.

Semantics of WDPTs. We define the semantics of WDPTs by naturally extending their interpretation under semantic web vocabularies [17, 19]. The intuition behind the semantics of a WDPT $(T, \lambda, \bar{x})$ is as follows. Each subtree T' of T rooted in r describes a pattern, namely CQ $q_{T'}$. A mapping h satisfies (T, λ) over a database $\mathcal{D}$, if it is "maximal" among the mappings that satisfy the patterns defined by the subtrees of T. This means, h satisfies the pattern defined by some subtree T' of T, and there is no way to "extend" h to satisfy the pattern of a bigger subtree T'' of T. The evaluation of WDPT $(T, \lambda, \bar{x})$ over $\mathcal{D}$ corresponds then to the projection over the variables in $\bar{x}$ of the mappings h that satisfy (T, λ) over $\mathcal{D}$. We formalize this next.

Definition 2. (Semantics of WDPTs) Let us consider a WDPT $p = (T, \lambda, \bar{x})$ and a database $\mathcal{D}$ over σ.

- A *homomorphism* from p to $\mathcal{D}$ is a partial mapping $h : \mathbf{X} \to \mathbf{U}$, for which it is the case that there is a subtree T' of T rooted in r (the distinguished root node of T) such that $h \in q_{T'}(\mathcal{D})$.
- The homomorphism h is *maximal* if there is no homomorphism h' from p to $\mathcal{D}$ such that $h \sqsubset h'$.

The *evaluation* of WDPT $p = (T, \lambda, \bar{x})$ over $\mathcal{D}$, denoted $p(\mathcal{D})$, corresponds to all mappings of the form $h_{\bar{x}}$, such that h is a maximal homomorphism from p to $\mathcal{D}$.

Notice that WDPTs properly extend CQs. In fact, assume $q(\bar{x})$ is a CQ of the form $\text{Ans}(\bar{x}) \leftarrow R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)$. Then $q(\bar{x})$ is equivalent to WDPT $p = (T, \lambda, \bar{x})$, where T consists of a single node r and $\lambda(r) = \{R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)\}$. In other words, $q(\mathcal{D}) = p(\mathcal{D})$, for each database $\mathcal{D}$. We typically do not distinguish between a CQ and the single node WDPT that represents it. On the other hand, as illustrated in Example 1, WDPTs express interesting properties that cannot be expressed as CQs.

RDF well-designed pattern trees. By the nature of semantic web vocabularies, WDPTs are defined in such context over a schema that consists of a single ternary relation. We call these *RDF WDPTs*. As recalled above, RDF WDPTs are equal in expressive power to well-designed SPARQL restricted to the AND and OPT operators [17, 18]. All lower bounds obtained in our paper can be proven to hold even for RDF WDPTs. Hence, all our results continue to hold in the RDF scenario.

3. EFFICIENT EVALUATION OF WDPTS

In this section we study the complexity of the evaluation problem for different classes $\mathcal{C}$ of WDPTs. This problem is formally defined as follows:

PROBLEM :	EVAL($\mathcal{C}$).
INPUT :	A database $\mathcal{D}$ and a WDPT $p \in \mathcal{C}$ over σ, and a partial mapping $h : \mathbf{X} \to \mathbf{U}$.
QUESTION :	Is $h \in p(\mathcal{D})$?

The complexity of EVAL($\mathcal{C}$) has been studied for the case when $\mathcal{C}$ is the class $\mathcal{C}_{\mathsf{all}}$ of all WDPTs or the class $\mathcal{C}_{\mathsf{pf}}$ of projection-free WDPTs. This is summarized next:

THEOREM 1. *The following hold:*

1. EVAL($\mathcal{C}_{\mathsf{all}}$) *is* Σ_2^P*-complete [17].*
2. EVAL($\mathcal{C}_{\mathsf{pf}}$) *is* coNP*-complete [18].*

That is, the evaluation problem is intractable (coNP-hard) even for the simple class of projection-free WDPTs. For the class of all WDPTs the complexity jumps to the second-level of the polynomial hierarchy. This raises the need for understanding which classes of WDPTs can be evaluated in polynomial time.

Evaluation of WDPTs is defined in terms of CQ evaluation, which is an intractable problem in general. Therefore, our goal of identifying tractable classes of WDPTs naturally calls for a restriction of the classes of CQ patterns allowed in them. This idea has already been successfully applied for obtaining tractable classes of projection-free WDPTs [17]. Extending this to the class of WDPTs with projection requires new conditions, which we develop in this section. It is important first, however, to review some of the classes of CQs that can be evaluated efficiently.

3.1 Tractable evaluation for CQs

The evaluation problem for a class $\mathcal{C}$ of CQs, denoted CQ-EVAL($\mathcal{C}$), is defined analogously to the case of WDPTs. That is, CQ-EVAL($\mathcal{C}$) is the problem of checking if $h \in q(\mathcal{D})$, given a database $\mathcal{D}$, a CQ $q \in \mathcal{C}$ and a partial mapping $h : \mathbf{X} \to \mathbf{U}$.

It is known that, without further restrictions, the evaluation problem for CQs is intractable; in particular, CQ-EVAL($\mathcal{C}$) is NP-complete when $\mathcal{C}$ is the class of all CQs [7]. Due to a myriad of papers in the last two decades, we have by now a very good understanding of which classes of CQs admit tractable evaluation. In this work, we concentrate on two of the most fundamental tractable classes of CQs: the class of CQs of bounded *treewidth* [8] and of bounded *hypertreewidth* [13], respectively, which are defined next.

CQs of bounded treewidth. A tractable class of CQs can be obtained by restricting the *treewidth* of the *hypergraph* of queries [8]. A hypergraph H is a pair (V, E), where V is a finite set of nodes and E is a finite set of *hyperedges*, i.e., subsets of V.

A *tree decomposition* of a hypergraph $H = (V, E)$ is a pair (S, ν), where S is a tree and $\nu : S \to 2^V$, that satisfies the

following: (1) For each $u \in V$ the set $\{s \in S \mid u \in \nu(s)\}$ is a connected subset of S, and (2) each hyperedge of E is contained in one of the sets $\nu(s)$, for $s \in S$. The *width* of (S, ν) is $(\max\{|\nu(s)| \mid s \in S\}) - 1$. The *treewidth* of H is the minimum width of its tree decompositions. Intuitively, the treewidth of H measures its *tree-likeness*. If H is an undirected graph, then H is acyclic iff it is of treewidth one.

Let q be the CQ $\text{Ans}(\bar{x}) \leftarrow R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)$. Its underlying hypergraph H_q is the pair (V, E), where V is the set of variables mentioned in q and E consists precisely of the sets of variables in the atoms $R_i(\bar{v}_i)$, for $1 \leq i \leq m$. For example, for the CQ $\text{Ans}() \leftarrow R(x, y, z), R(x, v, v), E(v, z)$, the hyperedges are $\{x, y, z\}$, $\{x, v\}$, and $\{v, z\}$. The treewidth of CQ q is the treewidth of H_q. We denote by $\mathsf{TW}(k)$ the class of CQs of treewidth at most k, for $k \geq 1$.

Example 4. Consider the CQ $\text{Ans}() \leftarrow E(x_1, x_2), \ldots, E(x_{n-1}, x_n)$ for $n \geq 3$. This CQ is in $\mathsf{TW}(1)$, since its hypergraph is a path, and, thus, acyclic. Adding the atom $E(x_1, x_n)$ increases the treewidth to two. Adding all atoms of the form $E(x_i, x_j)$, for $1 \leq i, j \leq n$, yields a CQ whose hypergraph is a clique of size n. Its treewidth is $n - 1$.

It follows from [8] (see also [10]) that evaluating CQs in $\mathsf{TW}(k)$, for $k \geq 1$, is a tractable problem:

THEOREM 2. *Let $k \geq 1$. Then* CQ-EVAL($\mathsf{TW}(k)$) *can be solved in* PTIME.

CQs of bounded hypertreewidth. The notion of treewidth is too restrictive when the arity of the schemas is not fixed in advance. In order to overcome this limitation, Gottlob et al. [13] proposed studying syntactic restrictions of the class of CQs based on *hypertree decompositions* of their hypergraphs. The analogue of treewidth in this context is the notion of hypertreewidth, which, like the former, leads to tractability of query evaluation.

A *hypertree decomposition* of a hypergraph $H = (V, E)$ is a triple (S, ν, κ), where S is a tree, ν is a map from S to 2^V, and κ is a map from S to 2^E, such that:

1. (S, ν) is a tree decomposition of H.
2. $\nu(s) \subseteq \bigcup \kappa(s)$ holds for every $s \in S$.

The *width* of (S, ν, κ) is defined as $\max_{s \in S} |\kappa(s)|$. The *hypertreewidth* of a hypergraph is the minimum width over all its hypertree decompositions.

The hypertreewidth of q is the hypertreewidth of H_q. We denote by $\mathsf{HW}(k)$ the class of all CQs with hypertreewidth at most k. Notably, $\mathsf{HW}(1)$ corresponds to the well-studied class AC of *acyclic* CQs [21]. Moreover, bounded treewidth is subsumed by bounded hypertreewidth; in particular, $\mathsf{TW}(k) \subseteq \mathsf{HW}(k+1)$, for every $k \geq 1$ [1]. On the other hand, as the next example shows, even $\mathsf{HW}(1) = \mathsf{AC}$ is not subsumed by any of the $\mathsf{TW}(k)$'s.

Example 5. Consider a class $\mathcal{C} = \{\theta_n \mid n \geq 2\}$ of CQs, where $\theta_n := \text{Ans}() \leftarrow \bigwedge_{1 \leq i < j \leq n} E(x_i, x_j), T_n(x_1, \ldots, x_n)$. It is easy to show that every CQ $\theta_n \in \mathcal{C}$ is in AC. On the other hand, the treewidth of the CQs in $\mathcal{C}$ is not bounded by any constant.

Evaluation of CQs of bounded hypertreewidth is not only polynomial but can be solved in the parallelizable complexity class LOGCFL, that lies in between NL and AC_1. Formally, this corresponds to the class of languages that can be reduced in logarithmic space to a context free language.

THEOREM 3. *[13] The problem* CQ-EVAL($\mathsf{HW}(k)$) *is complete for* LOGCFL *under logspace reductions, for every $k \geq 1$.*

Notice that this improves over the bound in Theorem 2 for the classes of CQs of bounded treewidth.

Remark. For historical reasons, hypertree decompositions are called *generalized* hypertree decompositions in the literature, and, correspondingly, hypertreewidth is known as generalized hypertreewidth [14]. Hypertreewidth is then obtained by imposing an extra condition on generalized hypertree decompositions. This condition ensures the tractability of the *recognizability* problem, i.e., determining if a hypergraph is of hypertreewidth k, for a fixed $k \geq 1$. For us, it is convenient to work with the more general and intuitive notion of hypertreewidth defined above.

3.2 Tractable evaluation of WDPTs

We now return to the main question of this section: When is the evaluation of WDPTs tractable? A condition that has been shown to help identifying relevant tractable fragments of WDPTs is *local tractability* [17]. This refers to restricting the CQ defined by each node in a WDPT to belong to a tractable class.

- **Local tractability:** Let $\mathcal{C}$ be a class of CQs for which CQ-EVAL($\mathcal{C}$) is tractable. A WDPT $(T, \lambda, \bar{x})$ is *locally in* $\mathcal{C}$, if for each node $t \in T$ such that $\lambda(t) = \{R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)\}$ the CQ $\text{Ans}() \leftarrow R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)$ is in $\mathcal{C}$.

  We write ℓ-$\mathcal{C}$ for the set of all WDPTs that are locally in $\mathcal{C}$.

It is known that local tractability leads to tractability of evaluation for projection-free WDPTs:

THEOREM 4. *[17] Let $\mathcal{C}$ be a class of CQs such that* CQ-EVAL($\mathcal{C}$) *is in* PTIME, *and assume $\mathcal{C}'$ is the class of projection-free WDPTs in ℓ-$\mathcal{C}$. Then* EVAL($\mathcal{C}'$) *is in* PTIME.

On the other hand, this result does not hold in the presence of projection, even when $\mathcal{C}$ is of bounded treewidth:

THEOREM 5. *[17]* EVAL(ℓ-$\mathsf{TW}(k)$) *and* EVAL(ℓ-$\mathsf{HW}(k)$) *are* NP*-complete for every $k \geq 1$.*

This raises the question of which further restrictions on WDPTs are needed to achieve tractability. Here we identify a natural such restriction, called *bounded interface*. Intuitively, this restricts the number of variables shared between a node in a WDPT and its children.

- **Bounded interface:** Let $c \geq 1$. A WDPT $(T, \lambda, \bar{x})$ has *c-bounded interface*, if for each node $t \in T$ with children $t_1, \ldots, t_k$ it is the case that the number of variables that appear both in a relational atom in $\lambda(t)$ and in a relational atom in $\lambda(t_i)$, for some $1 \leq i \leq k$, is at most c.

  We denote $\mathsf{BI}(c)$ the set of WDPTs of c-bounded interface.

Example 6. Let p be the WDPT from Figure 1. Then $p \in \ell$-$\mathsf{TW}(1)$ and $p \in \mathsf{BI}(2)$: Since each node contains exactly two variables, the treewidth of each node is trivially

1. Concerning the number of shared variables, observe that x occurs in both the root node and its first child, while y occurs in the root node and its second child. WDPT p thus has a 2-bounded interface.

Notice that the effect of bounding the interface of each node in a WDPT $(T, \lambda, \bar{x})$ is a restriction on the shape of the CQs of the form $q_{T'}$ (for every subtree T' of T rooted in r) that define the semantics of $(T, \lambda, \bar{x})$. In particular, c-bounded interface implies that the number of variables shared between two atoms $R(\bar{v})$ and $R'(\bar{v}')$ in $q_{T'}$ that come from different nodes of T' is at most c. Interestingly, similar restrictions on the number of variables shared by different atoms of CQs have been recently applied for obtaining reasonable bounds for the problem of containment of Datalog into unions of CQs [6].

Our main result of the section states that local tractability and bounded interface yield tractability of WDPT evaluation:

THEOREM 6. *Let $\mathcal{C}$ be a class of CQs for which* CQ-EVAL($\mathcal{C}$) *is in* PTIME *and $c \geq 1$ a positive integer. Then* EVAL(ℓ-$\mathcal{C} \cap \mathsf{BI}(c)$) *is also in* PTIME.

Recall that the evaluation problem for the CQ classes $\mathsf{TW}(k)$ and $\mathsf{HW}(k)$, for $k \geq 1$, is not only tractable but can be solved in the parallelizable class LOGCFL. In fact, the PTIME-algorithm for WDPT-evaluation in the proof of Theorem 6 can be refined to a LOGCFL-algorithm, provided that the corresponding CQ-evaluation problem is in LOGCFL. We thus obtain the following:

THEOREM 7. *Let $\mathcal{C}$ be a class of CQs for which* CQ-EVAL($\mathcal{C}$) *is in* LOGCFL *and $c \geq 1$ a positive integer. Then* EVAL(ℓ-$\mathcal{C} \cap \mathsf{BI}(c)$) *is also in* LOGCFL. *In particular,* EVAL(ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$) *and* EVAL($\ell$-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$) *are in* LOGCFL *for each $k, c \geq 1$.*

Notice that CQs can be considered as special case of WDPTs consisting of the root node only. Hence, $\mathsf{TW}(k) \subseteq \ell$-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$ and $\mathsf{HW}(k) \subseteq \ell$-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$ hold for each $c \geq 1$. Therefore, Theorem 7 tells us that ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$ and ℓ-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$ define relevant extensions of $\mathsf{TW}(k)$ and $\mathsf{HW}(k)$, respectively, that do not increase the complexity of evaluation. It follows from [12] that both EVAL(ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$) and EVAL($\ell$-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$) are LOGCFL-hard under logspace reductions.

3.3 Partial evaluation of WDPTs

Given the nature of WDPTs, it is also interesting to check whether a mapping h is a *partial* answer to the WDPT p over $\mathcal{D}$ [18], i.e., whether h can be "extended" to some answer h' to p over $\mathcal{D}$. This gives rise to the partial evaluation problem for $\mathcal{C}$ defined as follows.

PROBLEM :	PARTIAL-EVAL($\mathcal{C}$).
INPUT :	A database $\mathcal{D}$ and a WDPT $p \in \mathcal{C}$ over σ, and a partial mapping $h : \mathbf{X} \to \mathbf{U}$.
QUESTION :	Is there $h' \in p(\mathcal{D})$ such that $h \sqsubseteq h'$?

Partial evaluation is tractable for the class of projection-free WDPTs [18]. In contrast, if projection is allowed, then partial evaluation is intractable even under local tractability:

PROPOSITION 1. *[17]* PARTIAL-EVAL(ℓ-$\mathsf{TW}(k)$) *is* NP-*complete for every $k \geq 1$.*

Recall from Theorem 7 that the conjunction of local tractability and bounded interface leads to efficient (exact) evaluation of WDPTs. It is easy to modify the proof of Theorem 7 to show that also PARTIAL-EVAL(ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$) and PARTIAL-EVAL($\ell$-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$) are in LOGCFL. However, partial evaluation is seemingly easier than exact evaluation. Hence, the question naturally arises if tractability of partial evaluation of WDPTs can be ensured by a weaker condition. Indeed, we give a positive answer to this question below. This condition will be referred to as *global tractability*. Intuitively, it states that there is a bound on the treewidth (resp., hypertreewidth) of the CQs defined by the different subtrees of a WDPT $(T, \lambda, \bar{x})$ rooted in r.

- **Global tractability:** Let $\mathcal{C}$ be $\mathsf{TW}(k)$ or $\mathsf{HW}(k)$, for $k \geq 1$. A WDPT $(T, \lambda, \bar{x})$ is *globally in $\mathcal{C}$*, if for each subtree T' of T rooted in r it is the case that the CQ $q_{T'}$ is in $\mathcal{C}$.

  We denote with g-$\mathcal{C}$ the set of all WDPTs that are globally in $\mathcal{C}$.

The following proposition formally states that global tractability is a strictly weaker condition than the conjunction of local tractability and bounded interface. The first part of the proposition shows that local tractability plus bounded interface imply global tractability, while the second part shows that the opposite is not the case:

PROPOSITION 2. *The following hold:*

1. *Let $k, c \geq 1$. Then:*
   - ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c) \subseteq g$-$\mathsf{TW}(k + 2c)$.
   - ℓ-$\mathsf{HW}(k) \cap \mathsf{BI}(c) \subseteq g$-$\mathsf{HW}(k + 2c)$.
2. *For every $k \geq 1$ there is a family C_k of WDPTs in g-$\mathsf{TW}(k)$ (resp., in g-$\mathsf{HW}(k)$) such that $C_k \not\subseteq \mathsf{BI}(c)$, for each $c \geq 1$.*

We now formally prove that global tractability leads to tractability of the partial evaluation problem for WDPTs:

THEOREM 8. PARTIAL-EVAL(g-$\mathsf{TW}(k)$) *and* PARTIAL-EVAL(g-$\mathsf{HW}(k)$) *are in* LOGCFL *for every $k \geq 1$.*

It remains to answer the question if global tractability also suffices to ensure tractability of (exact) evaluation for WDPTs. Below we show that this is not the case.

PROPOSITION 3. EVAL(g-$\mathsf{TW}(k)$) *and* EVAL(g-$\mathsf{HW}(k)$) *are* NP-*complete for every $k \geq 1$.*

3.4 Semantics based on maximal mappings

The semantics of projection-free WDPTs is only based on *maximal* mappings, i.e., mappings that are not subsumed by any other mapping in the answer. This is no longer the case in the presence of projection as it has already been shown in Example 2 in the introduction.

Recent work on query answering for SPARQL under entailment regimes has established the need for a semantics for WDPTs that is uniquely based on maximal mappings [2]. This semantics is formalized as follows. Assume $\mathcal{D}$ is a

database and p is a WDPT over σ. The *evaluation of p over $\mathcal{D}$ under maximal mappings*, denoted $p_m(\mathcal{D})$, corresponds to the restriction of $p(\mathcal{D})$ to those mappings $h \in p(\mathcal{D})$ that are maximal with respect to $\sqsubseteq$.

Example 7. Let p be the WDPT from Figure 1, but assume the answers to be projected to $\bar{x} = \{y, z\}$, and $\mathcal{D}$ the database from Example 2. Then $p(\mathcal{D}) = \{\mu_1, \mu_2\}$ with $\mu_1(y) = \mu_2(y) =$ "*Caribou*" and $\mu_2(z) =$ "2", while $p_m(\mathcal{D}) = \{\mu_2\}$.

This naturally leads to the following decision problem:

PROBLEM :	MAX-EVAL($\mathcal{C}$).
INPUT :	A database $\mathcal{D}$ and a WDPT $p \in \mathcal{C}$ over σ, and a partial mapping $h : \mathbf{X} \to \mathbf{U}$.
QUESTION :	Is $h \in p_m(\mathcal{D})$?

It follows from [2] that MAX-EVAL($\mathcal{C}$) is intractable when $\mathcal{C}$ is the class of all WDPTs (more precisely, this problem is complete for the class DP, i.e., the class of languages that correspond to the intersection of a language in NP and one in coNP). To obtain tractability in this case it is sufficient to impose global tractability, which is exactly the same condition that yields tractability of partial evaluation for WDPTs (as stated in Theorem 8):

THEOREM 9. MAX-EVAL(g-$\mathsf{TW}(k)$) *and* MAX-EVAL(g-$\mathsf{HW}(k)$) *are in* LOGCFL *for every* $k \geq 1$.

Analogously to PARTIAL-EVAL, local tractability is not sufficient to ensure tractability of MAX-EVAL:

PROPOSITION 4. *For every* $k \geq 1$ *the problems* MAX-EVAL(ℓ-$\mathsf{TW}(k)$) *and* MAX-EVAL(ℓ-$\mathsf{HW}(k)$) *are* DP*-complete.*

4. CONTAINMENT AND SUBSUMPTION

Query containment and query equivalence are among the most fundamental problems in static query analysis, i.e., given two queries q_1 and q_2, one wants to test if – for any database $\mathcal{D}$ – the condition $q_1(\mathcal{D}) \subseteq q_2(\mathcal{D})$ or $q_1(\mathcal{D}) = q_2(\mathcal{D})$, respectively, holds. If this is the case, we write $q_1 \subseteq q_2$ or $q_1 \equiv q_2$, respectively. For CQs, these problems are NP-complete in the general case [7] and LOGCFL-complete if we restrict the CQs to one of the classes $\mathsf{TW}(k)$ or $\mathsf{HW}(k)$ [8, 13].

A detailed study of containment and equivalence of RDF WDPTs was carried out in [19]. In sharp contrast to the case of CQs, it was shown that both problems are undecidable. Now the question remains if the restriction to tractable fragments of WDPT evaluation can help. An inspection of the undecidability proofs in [19] shows that this is not the case. We thus get:

THEOREM 10 (IMPLICIT IN [19]). *The containment and equivalence problems of WDPTs are undecidable. The undecidability holds even if both WDPTs are from ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$ for arbitrary $k \geq 1$ and appropriately chosen constant c.*

In [3], it was observed that query containment of WDPTs may display an unintuitive behavior. Consequently, *subsumption* is proposed as a variant of containment: a WDPT p_1 is subsumed by p_2 (written as $p_1 \sqsubseteq p_2$) if, for every database $\mathcal{D}$, every answer $h \in p_1(\mathcal{D})$ is subsumed by an answer $h' \in p_2(\mathcal{D})$ [3]. Additionally, we define *subsumption-equivalence* (denoted as $p_1 \equiv_s p_2$) if both $p_1 \sqsubseteq p_2$ and $p_2 \sqsubseteq p_1$ hold. We thus study the following problems.

PROBLEM :	SUBSUMPTION($\mathcal{C}_1, \mathcal{C}_2$).
INPUT :	Two WDPTs $p_1 \in \mathcal{C}_1$ and $p_2 \in \mathcal{C}_2$ over σ.
QUESTION :	Does $p_1 \sqsubseteq p_2$ hold?

PROBLEM :	$\sqsubseteq$-EQUIVALENCE($\mathcal{C}_1, \mathcal{C}_2$).
INPUT :	Two WDPTs $p_1 \in \mathcal{C}_1$ and $p_2 \in \mathcal{C}_2$ over σ.
QUESTION :	Does $p_1 \equiv_s p_2$ hold?

In [17], the Π_2^P-completeness of SUBSUMPTION($\mathcal{C}_1, \mathcal{C}_2$) was proved where $\mathcal{C}_1$ and $\mathcal{C}_2$ denote the class of arbitrary WDPTs. It was also shown that Π_2^P-hardness holds even if we restrict both $\mathcal{C}_1$ and $\mathcal{C}_2$ to projection-free WDPTs.

The problem $\sqsubseteq$-EQUIVALENCE($\mathcal{C}_1, \mathcal{C}_2$) has not been studied so far. However, in [2] a closely related problem based on the "'maximal mappings" semantics from Section 3.4 was studied – the so-called MAXEQUIVALENCE($\mathcal{C}_1, \mathcal{C}_2$)-problem: Given two WDPTs $p \in \mathcal{C}_1$, $p' \in \mathcal{C}_2$, does $p_m(\mathcal{D}) = p'_m(\mathcal{D})$ hold for every database $\mathcal{D}$? In other words, we check if two WDPTs p and p' have the same *maximal* solutions over any database $\mathcal{D}$. If this is the case, we write $p \equiv_{\max} p'$. This problem was shown to be Π_2^P-complete in [2]. An inspection of the proof in [2] shows that Π_2^P-hardness holds even if one of the classes $\mathcal{C}_i$ is restricted to ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$ with $k = c = 2$. Below, we show that $\sqsubseteq$-EQUIVALENCE($\mathcal{C}_1, \mathcal{C}_2$) and MAXEQUIVALENCE($\mathcal{C}_1, \mathcal{C}_2$) are equivalent problems. In this way, we establish Π_2^P-completeness also for $\sqsubseteq$-EQUIVALENCE($\mathcal{C}_1, \mathcal{C}_2$).

PROPOSITION 5. *For any classes $\mathcal{C}_1, \mathcal{C}_2$ of WDPTs, the problems* MAXEQUIVALENCE*($\mathcal{C}_1, \mathcal{C}_2$) and* $\sqsubseteq$-EQUIVALENCE*($\mathcal{C}_1$, $\mathcal{C}_2$) are equivalent, i.e., for all $p \in \mathcal{C}_1$ and $p' \in \mathcal{C}_2$, we have*

$$p \equiv_s p' \quad \Leftrightarrow \quad p \equiv_{\max} p'.$$

We then immediately obtain the following:

COROLLARY 1. *Let $\mathcal{C}_1, \mathcal{C}_2$ be the class of arbitrary WDPTs. Then the problem* $\sqsubseteq$-EQUIVALENCE*($\mathcal{C}_1, \mathcal{C}_2$) is Π_2^P-complete. It remains Π_2^P-hard even if one of the classes $\mathcal{C}_i$ is restricted to ℓ-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$ (or to ℓ-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$) with $k = c = 2$.*

Now the natural question is if the restriction of $\mathcal{C}_1, \mathcal{C}_2$ to tractable classes of WDPT evaluation also leads to a lower complexity of SUBSUMPTION($\mathcal{C}_1, \mathcal{C}_2$) and $\sqsubseteq$-EQUIVALENCE($\mathcal{C}_1$, $\mathcal{C}_2$). This question is answered below.

THEOREM 11. SUBSUMPTION*($\mathcal{C}_1, \mathcal{C}_2$) is* coNP*-complete for the following classes $\mathcal{C}_1, \mathcal{C}_2$ of WDPTs:*

1. coNP*-membership holds even if $\mathcal{C}_1$ is the class of arbitrary WDPTs and $\mathcal{C}_2 \subseteq g$-$\mathsf{HW}(k)$ for any $k \geq 1$.*
2. coNP*-hardness holds even if $\mathcal{C}_1, \mathcal{C}_2 \subseteq \ell$-$\mathsf{HW}(k) \cap \mathsf{BI}(c)$ (or, likewise, if $\mathcal{C}_1, \mathcal{C}_2 \subseteq \ell$-$\mathsf{TW}(k) \cap \mathsf{BI}(c)$) holds for any $k \geq 1$ and $c = 1$.*

Membership is proved using techniques from [17]. Hardness uses a straightforward reduction from VALIDITY. We next give a similar complexity classification for $\sqsubseteq$-EQUIVALENCE.

THEOREM 12. *$\sqsubseteq$-EQUIVALENCE$(\mathcal{C}_1, \mathcal{C}_2)$ is* coNP*-complete for the following classes $\mathcal{C}_1, \mathcal{C}_2$ of WDPTs:*

1. coNP*-membership holds even if $\mathcal{C}_1, \mathcal{C}_2 \subseteq g\text{-}\mathsf{HW}(k)$ for any $k \geq 1$.*
2. coNP*-hardness holds even if $\mathcal{C}_1, \mathcal{C}_2 \subseteq \ell\text{-}\mathsf{HW}(k) \cap \mathsf{BI}(c)$ (or, likewise, if $\mathcal{C}_1, \mathcal{C}_2 \subseteq \ell\text{-}\mathsf{TW}(k) \cap \mathsf{BI}(c)$) holds for any $k \geq 1$ and $c = 2$.*

PROOF SKETCH. Membership follows from the coNP-membership of SUBSUMPTION$(\mathcal{C})$ in Theorem 11. Hardness is proved by a reduction from VALIDITY. To ensure $p_1 \equiv_s p_2$ in case of a valid formula ϕ, we need an involved construction. In particular, the selection of a particular truth assignment for the variables in X is encoded by the selection of $2m$ (with $m = |X|$) descendants of the root of p_2 from a collection of $3m$ possible descendants, which are arranged in a subtree of depth m. $\square$

Theorems 11 and 12 together with the Π_2^P-completeness results of [17] and Corollary 1 have left a small gap: What if both $\mathcal{C}_1$ and $\mathcal{C}_2$ are locally tractable classes? We close this gap below.

PROPOSITION 6. *The problems* SUBSUMPTION$(\mathcal{C}_1, \mathcal{C}_2)$ *and* $\sqsubseteq$-EQUIVALENCE$(\mathcal{C}_1, \mathcal{C}_2)$ *remain Π_2^P-complete even if both $\mathcal{C}_1$ and $\mathcal{C}_2$ are restricted to $\ell\text{-}\mathsf{HW}(k)$ or to $\ell\text{-}\mathsf{TW}(k)$ with $k \geq 2$.*

5. SEMANTIC OPTIMIZATION OF WDPTS

In Section 3, we developed conditions that lead to tractability for several variants of the WDPT evaluation problem. In this section, we study the *semantic space* defined by these conditions; that is, the space of WDPTs that are equivalent to a WDPT in a class syntactically defined via treewidth or hypertreewidth.

First we have to fix the right notion of *equivalence*. By Theorem 10 we know that strict equivalence ("$\equiv$") is undecidable even for the most restricted fragments of WDPTs considered here. Hence, we have to be contented with a relaxed notion of equivalence – *subsumption equivalence* ("$\equiv_s$") introduced above.

But then we also have to choose the appropriate variant of WDPT evaluation: subsumption equivalence preserves *partial* and *maximal* solutions. Hence, we shall focus on the PARTIAL-EVAL$(\mathcal{C})$ and MAX-EVAL$(\mathcal{C})$ problems here. It should be noted that the MAX-EVAL, PARTIAL-EVAL and EVAL problems coincide for CQs, i.e., WDPTs consisting of the root node only.

Finally, we determine the right syntactical restriction on WDPTs to ensure tractability of these problems. By Theorems 8 and 9, the restriction to $g\text{-}\mathsf{TW}(k)$ or $g\text{-}\mathsf{HW}(k)$ for constant k is sufficient. At this point, the discussion of a significant difference between treewidth and hypertreewidth is in order: It will turn out convenient to choose our fragment of CQs in such a way that it is closed under taking *arbitrary* subqueries. While $\mathsf{TW}(k)$ enjoys this property, $\mathsf{HW}(k)$ does not. We therefore restrict $\mathsf{HW}(k)$ to the class $\mathsf{HW}'(k)$ consisting of all CQs q such that each subquery q' of q has hypertreewidth at most k. In [15], this restricted notion of hypertreewidth was called β-hypertreewidth in analogy with β-acyclicity introduced in [11]. We thus define the class

$$\mathsf{WB}(k) \;=\; g\text{-}\mathcal{C}(k), \quad \text{for } k \geq 1,$$

with either $\mathcal{C}(k) = \mathsf{TW}(k)$ or $\mathcal{C}(k) = \mathsf{HW}'(k)$. The acronym WB stands for *well-behaved*. Most results presented below hold for both choices of $\mathcal{C}(k)$. These results will thus simply be stated for $\mathsf{WB}(k)$ without distinguishing between $g\text{-}\mathsf{TW}(k)$ and $g\text{-}\mathsf{HW}'(k)$. However, there are also some results (in particular upper bounds) where the concrete choice of $\mathcal{C}(k)$ does make a difference in that an additional NP-oracle is needed in case of $\mathcal{C}(k) = g\text{-}\mathsf{HW}'(k)$. The oracle is used to verify that some WDPT indeed is in $g\text{-}\mathsf{HW}'(k)$. The problem here is that it is not known if, for given k, it can be can be tested efficiently if β-hypertreewidth $\leq k$ holds.

The semantic space defined by classes of the form $\mathsf{WB}(k)$ is formally defined below.

Definition 3. ($\mathcal{M}(\mathsf{WB}(k))$) Let $k \geq 1$. We denote by $\mathcal{M}(\mathsf{WB}(k))$ the class of WDPTs p for which there is a WDPT $p' \in \mathsf{WB}(k)$ such that $p \equiv_s p'$.

We show that these classes are decidable. We then apply this result to show that the partial and maximal evaluation problems for WDPTs in $\mathcal{M}(\mathsf{WB}(k))$ are fixed-parameter tractable (when taking the size of the WDPT as the parameter). This is an improvement with respect to the corresponding evaluation problems for arbitrary WDPTs and even for CQs. For the latter, no fixed-parameter tractable algorithm is believed to exist. Finally, we study the notion of $\mathsf{WB}(k)$-approximation for WDPTs.

5.1 Decidability of $\mathsf{WB}(k)$ modulo equivalence

We start by stating the decidability of our notion:

THEOREM 13. *Let $k \geq 1$. There is a* NEXPTIME^{NP} *algorithm that, given a WDPT p, decides if p is in $\mathcal{M}(\mathsf{WB}(k))$, and, if this is the case, constructs a WDPT p' in $\mathsf{WB}(k)$ of at most exponential size in p such that $p \equiv_s p'$. The* NP*-oracle is omitted if $\mathsf{WB}(k) = g\text{-}\mathsf{TW}(k)$.*

The proof of this result follows from the next lemma:

LEMMA 1. *Let p and p' be WDPTs such that $p' \sqsubseteq p$ and $p' \in \mathsf{WB}(k)$. Then there exists $p'' \in \mathsf{WB}(k)$ such that (1) $p' \sqsubseteq p'' \sqsubseteq p$, and (2) the size of p'' is at most exponential in the size of p.*

We now explain how Theorem 13 follows from Lemma 1. Assume p is in $\mathcal{M}(\mathsf{WB}(k))$, i.e., there is a WDPT p' in $\mathsf{WB}(k)$ such that $p \equiv_s p'$. Since $p' \sqsubseteq p$ and $p' \in \mathsf{WB}(k)$, we have from Lemma 1 that there is a WDPT $p'' \in \mathsf{WB}(k)$ such that (1) $p' \sqsubseteq p'' \sqsubseteq p$, and (2) the size of p'' is at most exponential in the size of p. We conclude that $p \equiv_s p''$ since $p \sqsubseteq p' \sqsubseteq p'' \sqsubseteq p$. Hence, if p is in $\mathcal{M}(\mathsf{WB}(k))$, then there is a WDPT p' in $\mathsf{WB}(k)$ with $p \equiv_s p'$ and the size of p' is at most exponential in the size of p. Then the NEXPTIME$^{\text{NP}}$ algorithm in Theorem 13 simply guesses such p' and checks (1) if $p' \in \mathsf{WB}(k)$ and (2) if it is subsumption-equivalent to p. Condition (1) requires an NP-oracle if $\mathsf{WB}(k) = g\text{-}\mathsf{HW}'(k)$. Condition (2) is satisfied if certain (exponentially many) homomorphisms exist [17]: they can be guessed alongside p' itself and do not increase the complexity.

While the upper bound in Theorem 13 might not be optimal, we can prove that the problem is at least on the second-level of the polynomial hierarchy:

PROPOSITION 7. *Let $k > 1$. The problem of checking whether a WDPT p belongs to $\mathcal{M}(\mathsf{WB}(k))$ is Π_2^P-hard.*

Notice that this establishes a difference with the analogous problem of checking whether a CQ is equivalent to one in a tractable class: For each $k \geq 1$, checking whether a CQ q is equivalent to some CQ q' in $\mathsf{TW}(k)$ or $\mathsf{HW}'(k)$ is in NP [10].

Evaluation for WDPTs in $\mathcal{M}(\mathsf{WB}(k))$. An important corollary of Theorem 13 is the following fixed-parameter tractability result:

COROLLARY 2. *Let $k \geq 1$. Then the problems* PARTIAL-EVAL($\mathcal{M}(\mathsf{WB}(k))$) *and* MAX-EVAL($\mathcal{M}(\mathsf{WB}(k))$) *are fixed-parameter tractable (when taking the size of the WDPT as the parameter).*

5.2 $\mathsf{WB}(k)$-Approximations of WDPTs

When a query q is not equivalent to one in a well-behaved class $\mathcal{Q}$, it might be useful to compute an *approximation* of q in $\mathcal{Q}$ [4]. Recall that this is a query $q' \in \mathcal{Q}$ that is *maximally contained* in q with respect to all queries in $\mathcal{Q}$. In other words, $q' \subseteq q$, and there is no $q'' \in \mathcal{Q}$ such that $q' \subset q'' \subseteq q$. For the reasons given before, we define approximations in the WDPT context not in terms of containment, but subsumption. Throughout this section we assume that WDPTs *do not contain constants*. The reason is that the notion of approximations with constants is problematic and not even well understood in the CQ context [4].

We now define approximations in the WDPT context. We write $p \sqsubset p'$ to denote that $p \sqsubseteq p'$ but $p \not\equiv_s p'$.

Definition 4. ($\mathsf{WB}(k)$-approximations) Let $k \geq 1$. Assume p and p' are WDPTs such that $p' \in \mathsf{WB}(k)$. Then p' is a $\mathsf{WB}(k)$-*approximation* of p if (1) $p' \sqsubseteq p$, and (2) there is no $p'' \in \mathsf{WB}(k)$ such that $p' \sqsubset p'' \sqsubseteq p$.

Existence of approximations. The most important question in the context of approximations is whether approximations always exist [4, 5]. The techniques developed in Lemma 1 allow us to prove that this is indeed the case in the WDPT scenario. Furthermore, for each WDPT, an exponential size approximation can be constructed in double-exponential time:

THEOREM 14. *Let $k \geq 1$. There is a double-exponential time algorithm that, given a WDPT p, constructs an exponential size $\mathsf{WB}(k)$-approximation p' of p.*

Complexity. In order to better understand the complexity of computing approximations, we study the following decision problem: Given WDPTs p and p' such that $p' \in \mathsf{WB}(k)$, for $k \geq 1$, is p' a $\mathsf{WB}(k)$-approximation of p? We call this problem $\mathsf{WB}(k)$-APPROXIMATION. The next proposition establishes some upper and lower bounds for the problem.

PROPOSITION 8. *The following hold:*

1. $\mathsf{WB}(k)$-APPROXIMATION *is in* coNEXPTIME^{NP} *for each $k \geq 1$. The* NP*-oracle is omitted if $\mathsf{WB}(k) = g\text{-}\mathsf{TW}(k)$.*

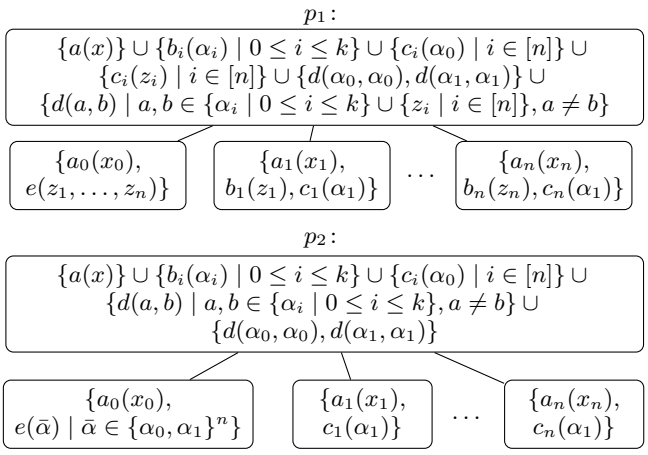

Figure 2: Exponential blow-up from p_1 to p_2.

2. $\mathsf{WB}(k)$-APPROXIMATION *is Π_2^P-hard for each $k > 1$.*
3. *If the input of the problem includes the promise that $p' \sqsubseteq p$, then* $\mathsf{WB}(k)$-APPROXIMATION *is Σ_2^P-hard for each $k > 1$.*

Again, this shows that our problem is harder than an analogous problem for CQs: For each $k \geq 1$, the problem of checking whether a CQ q is a $\mathsf{TW}(k)$-approximation of CQ q' is DP-complete. If, in addition, the input includes the promise that $q \subseteq q'$, then the problem becomes coNP-complete [4].

Size of approximations. We have seen above that approximations always exist even though Lemma 1 only allowed us to give an exponential upper bound on their size. The proof of that lemma centered around the properties of subsumption – without making use of the specific properties of approximations. One may thus ask if exponential size is indeed attainable by approximations. We give an affirmative answer to this question. This establishes another sharp contrast with CQs, where every $\mathsf{TW}(k)$-approximation is equivalent to one of polynomial size [4].

THEOREM 15. *For every $k \geq 2$, there exists a sequence of pairs of WDPTs $(p_1^{(n)}, p_2^{(n)})$, such that (1) $p_2^{(n)}$ is a* $\mathsf{WB}(k)$-approximation *of $p_1^{(n)}$, and (2) $p_2^{(n)}$ is necessarily* exponentially bigger *than $p_1^{(n)}$. More precisely, we have*

$$|p_1^{(n)}| = O(n^2) \quad \text{and} \quad |p_2^{(n)}| = \Omega(2^n),$$

and, for every WDPT $p_3^{(n)} \in \mathsf{WB}(k)$ with $p_2^{(n)} \sqsubseteq p_3^{(n)} \sqsubseteq p_1^{(n)}$, we have $|p_3^{(n)}| \geq |p_2^{(n)}|$.

PROOF SKETCH. Let $k \geq 1$ and $n \geq 1$. Consider the WDPTs $p_1^{(n)}$ and $p_2^{(n)}$ in Figure 2 with free variables $X = \{x, x_0, \ldots, x_n\}$. For the sake of readability, we omit superscript (n) from now on. Clearly, p_2 is exponentially bigger than p_1 and $p_2 \sqsubseteq p_1$ hold. Actually, we do not insist that p_2 is indeed a $\mathsf{WB}(k)$-approximation of p_1. It suffices to show that for every WDPT $p_3 \in \mathsf{WB}(k)$ with $p_2 \sqsubseteq p_3 \sqsubseteq p_1$, we have $|p_3| \geq |p_2|$.

We briefly discuss the main ideas of this construction. The WDPT p_1 is outside $\mathsf{WB}(k)$ due to a big clique of size $k{+}1{+}n$

(i.e., the d-atoms) in the root. The WDPT p_2 is obtained by instantiating all variables z_i to one of the two values α_0 or α_1. This is enforced by the fact that (by the condition $p_2 \in \mathsf{WB}(k)$) the big clique has to be shrunk to at most $k+1$ vertices and (by the condition $p_2 \sqsubseteq p_1$) there must exist a homomorphism from the root of p_1 to the root of p_2.

The variables z_i in the atom $e(z_1, \ldots, z_n)$ in the first leaf node of p_1 have to be instantiated in the same way. Again consider the condition $p_2 \sqsubseteq p_1$. As in the proof sketch of Lemma 1, this requires the existence of certain homomorphisms from subtrees of p_1 to subtrees of p_2. We thus consider every subtree p_2' of p_2 consisting of the root, the first leaf and an arbitrary subset of the remaining leaf nodes. We have to find a homomorphism from the corresponding subtree p_1' of p_1 (i.e., p_1' has the same free variables as p_2') into p_2'. It turns out that those variables z_i such that the leaf with atom $b_1(z_i)$ is contained in p_2' have to be mapped to α_1 while the other z_i's have to be mapped to α_0. Therefore, the first leaf of p_2 indeed has to contain all 2^n possible instantiations of atom $e(z_1, \ldots, z_n)$, resulting in an exponential blow-up. □

6. UNIONS OF WDPTS

Closing WDPTs under union constitutes one of the basic extensions of the language [18, 19]. Formally, a *union of WDPTs* (UWDPT) is an expression ϕ of the form $\bigcup_{1 \leq i \leq n} p_i$, where each p_i is a WDPT over σ. (Notice that we do not require different p_i's to have the same set of free variables). The evaluation of ϕ over database $\mathcal{D}$, denoted $\phi(\mathcal{D})$, corresponds to the set $\bigcup_{1 \leq i \leq n} p_i(\mathcal{D})$.

As before, we write $\phi \sqsubseteq \phi'$, for UWDPTs ϕ and ϕ', if for every database $\mathcal{D}$ and partial mapping $h \in \phi(\mathcal{D})$ it is the case that there is $h' \in \phi'(\mathcal{D})$ such that $h \sqsubseteq h'$. Similarly, we write $\phi \equiv_s \phi'$ whenever $\phi \sqsubseteq \phi'$ and $\phi' \sqsubseteq \phi$, and $\phi \sqsubset \phi'$ if $\phi \sqsubseteq \phi'$ but $\phi \not\equiv_s \phi'$.

If $\mathcal{C}$ is a class of WDPTs, we denote by $\bigcup$-Eval($\mathcal{C}$) the problem of determining if $h \in \phi(\mathcal{D})$, for ϕ a union of WDPTs in $\mathcal{C}$, $\mathcal{D}$ a database, and $h : \mathbf{X} \to \mathbf{U}$ a partial mapping. Similarly, we define $\bigcup$-Partial-Eval($\mathcal{C}$) and $\bigcup$-Max-Eval($\mathcal{C}$). It is immediate that unions of WDPTs from a well-behaved class $\mathcal{C}$ in terms of (variants of) evaluation preserve the good properties of $\mathcal{C}$:

Theorem 16. *The following hold for each $k \geq 1$ assuming $\mathcal{C}(k) = \mathsf{TW}(k)$ or $\mathsf{HW}(k)$:*

1. *The problem $\bigcup$-Eval(ℓ-$\mathcal{C}(k) \cap \mathsf{BI}(c)$) is in* LogCFL *for each $c \geq 1$.*
2. *$\bigcup$-Partial-Eval(g-$\mathcal{C}(k)$) and $\bigcup$-Max-Eval(g-$\mathcal{C}(k)$) are in* LogCFL.

In other words, the additional expressive power of UWDPTs compared with WDPTs has no effect on our variants of the evaluation problem. We look next at semantic optimization into well-behaved classes of UWDPTs. It will turn out that there the extension from WDPTs to UWDPTs makes a huge difference.

Semantic optimization of UWDPTs. Following Section 5, we concentrate on unions of WDPTs from $\mathsf{WB}(k) = g\text{-}\mathcal{C}(k)$, where $\mathcal{C}(k)$ is either $\mathsf{TW}(k)$ or $\mathsf{HW}'(k)$. We thus define:

$$\mathsf{UWB}(k) \;=\; \{ \bigcup_{1 \leq i \leq n} p_i \mid p_i \in \mathsf{WB}(k), \text{for each } 1 \leq i \leq n\}.$$

Analogously, we define the classes $\mathcal{M}(\mathsf{UWB}(k))$ of UWDPTs that are $\equiv_s$-equivalent to queries in $\mathsf{UWB}(k)$:

$$\mathcal{M}(\mathsf{UWB}(k)) \;=\; \{\phi \mid \phi \equiv_s \phi', \text{for some } \phi' \text{ in } \mathsf{UWB}(k)\}.$$

We prove below that the class $\mathcal{M}(\mathsf{UWB}(k))$ is not only decidable but allows for a nice characterization. To present this characterization, we introduce some useful notation first.

Given a WDPT $p = (T, \lambda, \bar{x})$ and a subtree T' of T rooted in r, we denote by $r_{T'}$ the CQ $\mathrm{Ans}(\bar{x}') \leftarrow R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)$, where $\{R_1(\bar{v}_1), \ldots, R_m(\bar{v}_m)\}$ is the set of all relational atoms in T' and $\bar{x}'$ is the set of variables that appear in $\bar{x}$ and in some $\bar{v}_i$, for $1 \leq i \leq m$. In other words, $r_{T'}$ is exactly as $q_{T'}$, only that we now take the projection over those variables from the $R_i(\bar{v}_i)$'s that appear free in p. Let us then define:

$$\phi_{\mathrm{cq}} \;=\; \bigcup_{p=(T,\lambda,\bar{x}) \in \phi} \;\; \bigcup_{T' \text{ a subtree of } T \text{ rooted in } r} r_{T'}.$$

It is not hard to see that $\phi \equiv_s \phi_{\mathrm{cq}}$ holds.

Example 8. Consider the RDF WDPT p introduced in Example 1, and the projection onto the variables $\{y, z, z'\}$ introduced in Example 3. When replacing the triple patterns by binary atoms, we get p_{cq} as the union of the following CQs:

- $\mathrm{Ans}(y) \leftarrow \texttt{rec\_by}(x, y), \texttt{publ}(x, \text{“after\_2010”})$.
- $\mathrm{Ans}(y, z) \leftarrow \texttt{rec\_by}(x, y), \texttt{publ}(x, \text{“after\_2010”}), \texttt{NME\_ranking}(x, z)$.
- $\mathrm{Ans}(y, z') \leftarrow \texttt{rec\_by}(x, y), \texttt{publ}(x, \text{“after\_2010”}), \texttt{formed\_in}(y, z')$.
- $\mathrm{Ans}(y, z, z') \leftarrow \texttt{rec\_by}(x, y), \texttt{publ}(x, \text{“after\_2010”}), \texttt{NME\_rating}(x, z), \texttt{formed\_in}(y, z')$.

Here we use `rec_by` and `publ` as abbreviation for `recorded_by` and `published`, respectively.

With ϕ_{cq} we have a useful tool for our further analysis of the class $\mathcal{M}(\mathsf{UWB}(k))$. In particular, this allows us to give the following characterization of $\mathcal{M}(\mathsf{UWB}(k))$.

Proposition 9. *Let $k \geq 1$. A UWDPT ϕ is in $\mathcal{M}(\mathsf{UWB}(k))$ iff ϕ_{cq} is $\sqsubseteq$-equivalent to a union of CQs in $\mathcal{C}(k)$.*

Applying Proposition 9 we obtain the following:

Theorem 17. *The following hold for each $k \geq 1$:*

1. *The problem of checking if a UWDPT ϕ is in $\mathcal{M}(\mathsf{UWB}(k))$ is in Π_2^P if $\mathcal{C}(k) = \mathsf{TW}(k)$, and in Π_3^P if $\mathcal{C}(k) = \mathsf{HW}'(k)$ For $k > 1$ either problem is Π_2^P-hard.*
2. *There is an* Exptime *algorithm that, given ϕ in $\mathcal{M}(\mathsf{UWB}(k))$, constructs a union ϕ' of (possibly exponentially many) WDPTs in $\mathsf{WB}(k)$ such that (1) each WDPT in ϕ' is of polynomial size, and (2) $\phi \equiv_s \phi'$.*

Notice the stark contrast of this result with the problem of checking whether a WDPT p is in $\mathcal{M}(\mathsf{WB}(k))$, for which we could only obtain a NExptime$^{\mathrm{NP}}$ upper bound in Theorem 13.

Evaluation for UWDPTs in $\mathcal{M}(\mathsf{UWB}(k))$. Analogously to the case of Corollary 2, it follows from Theorem 17 that the maximal and partial evaluation problems for queries in $\mathcal{M}(\mathsf{UWB}(k))$ are fixed-parameter tractable:

COROLLARY 3. *Let $k \geq 1$. Then* $\bigcup$-PARTIAL-EVAL($\mathcal{M}(\mathsf{UWB}(k))$) *and* $\bigcup$-MAX-EVAL($\mathcal{M}(\mathsf{UWB}(k))$) *are fixed-parameter tractable (when considering the size of the UWDPT as parameter).*

$\mathsf{UWB}(k)$-approximations. As in the case of Section 5.2, we study approximations for UWDPTs without constants. Fix $k \geq 1$. Let ϕ, ϕ' be UWDPTs such that $\phi' \in \mathsf{UWB}(k)$. Analogously to Definition 4, we have that ϕ' is a $\mathsf{UWB}(k)$-approximation of ϕ if (1) $\phi' \sqsubseteq \phi$, and (2) there is no UWDPT $\phi'' \in \mathsf{UWB}(k)$ such that $\phi' \sqsubset \phi'' \sqsubseteq \phi$.

The previous machinery allows us to develop a theory of approximations for UWDPTs. First of all, we can prove that approximations always exist and can be computed in exponential time. Second, approximations are unique up to $\equiv_s$-equivalence and consist of (possibly exponentially many) WDPTs of polynomial size (actually, these WDPTs are even CQs).

THEOREM 18. *There is an* EXPTIME *algorithm that, given a UWDPT ϕ, constructs a union ϕ' of (possibly exponentially many) WDPTs in $\mathsf{WB}(k)$ such that (1) each WDPT in ϕ' is of polynomial size, and (2) ϕ' is the unique (up to $\equiv_s$-equivalence) $\mathsf{UWB}(k)$-approximation of ϕ.*

These techniques also allow us to find reasonable bounds for the problem of checking if ϕ' is a $\mathsf{UWB}(k)$-approximation of ϕ. This problem is called $\mathsf{UWB}(k)$-APPROXIMATION.

PROPOSITION 10. *The problem* $\mathsf{UWB}(k)$-APPROXIMATION *is in Π_2^P if $\mathcal{C}(k) = \mathsf{TW}(k)$, and in Π_3^P if $\mathcal{C}(k) = \mathsf{HW}'(k)$. For $k \geq 1$ either problem is Π_2^P-hard.*

This is again in stark contrast with the problem of checking if a WDPT ϕ' is a $\mathsf{WB}(k)$-approximation of ϕ, for which we could only obtain a coNEXPTIME upper bound in Proposition 8.

7. CONCLUSION

In this work we have studied well-designed pattern trees (WDPTs) as a natural extension of conjunctive queries (CQs) by optional matching. We have considered WDPTs over arbitrary relational schemas here. However, all our results also apply to the corresponding fragment of the semantic web query language SPARQL by restricting the schema to a single ternary relation.

We have extended the search for tractable query evaluation and tractable query analysis from CQs to WDPTs. It has turned out that additional restrictions are required to ensure tractability of query evaluation of WDPTs. In Table 1, we give an overview of the complexities. The five rows refer to the five problems EVAL, PARTIAL-EVAL, MAX-EVAL, subsumption ($\sqsubseteq$), and subsumption-equivalence ($\equiv_s$). Completeness results are abbreviated with "c". The results marked with references are (at least implicitly) proved in previous work. Arrows indicate that the non-trivial part of these results carries over from the more special case ($\leftarrow$) or from the more general case ($\rightarrow$), respectively.

We have then applied our tractable classes of query evaluation to study semantic optimization and to initiate a theory of approximation of WDPTs. To this end, we have defined the classes $\mathsf{WB}(k)$ and $\mathsf{UWB}(k)$ of (unions) of "well-behaved" queries. Above all, we have managed to prove fixed-parameter tractability of query evaluation for (unions of) WDPTs that are $\equiv_s$-equivalent to a query in $\mathsf{WB}(k)$ or $\mathsf{UWB}(k)$, respectively. Further problems studied in this context are $\mathsf{WB}(k)$ / $\mathsf{UWB}(k)$-MEMBERSHIP (is a WDPT resp. a union of WDPTs $\equiv_s$-equivalent to a well-behaved one?) and $\mathsf{WB}(k)$ / $\mathsf{UWB}(k)$-APPROXIMATION (is a WDPT resp. a union of WDPTs an approximation of the other?). Preliminary complexity results for these tasks are displayed in Table 2. The upper bounds refer to the case $\mathsf{WB}(k) = g\text{-}\mathsf{HW}'(k)$. For $\mathsf{WB}(k) = g\text{-}\mathsf{TW}(k)$, the NP-oracle can be omitted and Π_3^P drops to Π_2^P, respectively.

Several lines of future work should be pursued. As far as query evaluation and query analysis are concerned, we yet have to identify a natural fragment of WDPTs that guarantees tractable subsumption and subsumption-equivalence. Towards a theory of semantic optimization of WDPTs, we have only made the first steps here. A better understanding of the nature of $\mathsf{WB}(k)$- and $\mathsf{UWB}(k)$-approximations is needed to close the gaps in Table 2. For instance, we conjecture that there always exists some approximation of polynomial size and that the complexity of $\mathsf{WB}(k)$-APPROXIMATION drops to the polynomial hierarchy. The situation of WDPTs is much more involved than for CQs, where the analogous problems come down to simple containment tests.

	general	l-$\mathcal{C}(k)$	g-$\mathcal{C}(k)$	l-$\mathcal{C}(k) \cap \mathsf{BI}(k)$
EVAL	Σ_2^P [17]	NP [17]	NP	LOGCFL
P-EVAL	$\leftarrow$	NP [17]	LOGCFL	$\rightarrow$
M-EVAL	$\leftarrow$	DP	LOGCFL	$\rightarrow$
$\sqsubseteq$	$\leftarrow$	Π_2^P	coNP	coNP
$\equiv_s$	$\leftarrow$	Π_2^P	coNP	coNP

Table 1: Complexity of WDPT evaluation and query analysis (all entries denote completeness).

	lower b.	upper bound
$\mathsf{WB}(k)$-MEMBERSHIP	Π_2^P	$\text{NEXPTIME}^{\text{NP}}$
$\mathsf{WB}(k)$-APPROXIMATION	Π_2^P	$\text{coNEXPTIME}^{\text{NP}}$
$\mathsf{UWB}(k)$-MEMBERSHIP	Π_2^P	Π_3^P
$\mathsf{UWB}(k)$-APPROXIMATION	Π_2^P	Π_3^P

Table 2: Semantic Optimization of WDPTs: lower- and upper bounds of the complexity.

Acknowledgments

The work of Pablo Barceló is funded by the Millenium Nucleus Center for Semantic Web Research under grant NC120004. Part of this work was done while Reinhard Pichler and Sebastian Skritek were visiting Pablo Barceló on invitation by the Millenium Nucleus Center for Semantic Web Research. Reinhard Pichler and Sebastian Skritek were supported by the Vienna Science and Technology Fund (WWTF) through project ICT12-015 and by the Austrian Science Fund (FWF):P25207-N23.

8. REFERENCES

[1] I. Adler, G. Gottlob, and M. Grohe. Hypertree width and related hypergraph invariants. *Eur. J. Comb.*, 28(8):2167–2181, 2007.

[2] S. Ahmetaj, W. Fischl, R. Pichler, M. Simkus, and S. Skritek. Towards reconciling SPARQL and certain answers. To appear in WWW 2015.

[3] M. Arenas and J. Pérez. Querying semantic web data with SPARQL. In *PODS*, pages 305–316, 2011.

[4] P. Barceló, L. Libkin, and M. Romero. Efficient approximations of conjunctive queries. *SIAM J. Comput.*, 43(3):1085–1130, 2014.

[5] P. Barceló, M. Romero, and M. Y. Vardi. Semantic acyclicity on graph databases. In *PODS*, pages 237–248, 2013.

[6] P. Barceló, M. Romero, and M. Y. Vardi. Does query evaluation tractability help query containment? In *PODS*, pages 188–199, 2014.

[7] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *STOC*, pages 77–90, 1977.

[8] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239(2):211–229, 2000.

[9] S. A. Cook and P. McKenzie. Problems complete for deterministic logarithmic space. *J. Algorithms*, 8(3):385–394, 1987.

[10] V. Dalmau, P. G. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *CP*, pages 310–326, 2002.

[11] R. Fagin. Degrees of acyclicity for hypergraphs and relational database schemes. *J. ACM*, 30(3):514–550, 1983.

[12] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *J. ACM*, 48(3):431–498, 2001.

[13] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *J. Comput. Syst. Sci.*, 64(3):579–627, 2002.

[14] G. Gottlob, Z. Miklós, and T. Schwentick. Generalized hypertree decompositions: NP-hardness and tractable variants. *J. ACM*, 56(6), 2009.

[15] G. Gottlob and R. Pichler. Hypergraphs in model checking: Acyclicity and hypertree-width versus clique-width. *SIAM J. Comput.*, 33(2):351–378, 2004.

[16] M. Grohe and D. Marx. Constraint solving via fractional edge covers. *ACM Transactions on Algorithms*, 11(1):4, 2014.

[17] A. Letelier, J. Pérez, R. Pichler, and S. Skritek. Static analysis and optimization of semantic web queries. *ACM Trans. Database Syst.*, 38(4):25, 2013.

[18] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. *ACM Trans. Database Syst.*, 34(3), 2009.

[19] R. Pichler and S. Skritek. Containment and equivalence of well-designed SPARQL. In *PODS'14*, pages 39–50, 2014.

[20] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. W3C Recommendation, Jan. 2008.

[21] M. Yannakakis. Algorithms for acyclic database schemes. In *VLDB*, pages 82–94, 1981.

APPENDIX

A. ADDITIONAL PROOF DETAILS

A.1 Proof (sketches) for Section 3

PROOF SKETCH OF THEOREM 6. Given a WDPT $p \in \ell\text{-}\mathcal{C} \cap \mathsf{BI}(c))$, a database $\mathcal{D}$ and a partial mapping h, the main idea of the polynomial time algorithm is to construct a Boolean acyclic CQ q on a new database $\mathcal{D}'$ such that $q(\mathcal{D}') = \mathit{true}$ iff $h \in p(\mathcal{D})$.

The CQ q is constructed from $p = (T, \lambda, \bar{x})$ as follows: Let $\bar{x}'$ be the subset of variables from $\bar{x}$ on which h is defined, and let T' be the minimal subtree of T that contains $\bar{x}'$ (and no more variables from $\bar{x}$). Moreover, let T'' be the *maximal* subtree of T that contains $\bar{x}'$ but no additional variable from $\bar{x}$. By the well-designedness of p, these subtrees are uniquely defined. For every node $t \in T'$, let $\bar{y}_t$ be the set of existentially quantified variables that appear both in $\lambda(t)$ and in $\lambda(t_i)$, for some child node t_i of t. We invent a new relation symbol R_t or arity $|\bar{y}_t|$ and define $q = \text{Ans}() \leftarrow \bigwedge_{t \in T'} R_t(\bar{y}_t)$.

Database $\mathcal{D}'$ is defined in two steps: First, for every node $t \in T'$, we compute all mappings g on the variables $\bar{y}_t$, such that $g \cup h$ can be extended to a homomorphism from $\lambda(t)$ into $\mathcal{D}$, i.e., g contains all instantiations of the (existentially quantified) "interface" variables consistent with solution candidate h. We define an intermediate database $\mathcal{D}''$ as the set of all atoms $R_t(g(\bar{y}))$. One can check that $\mathcal{D}''$ fulfills the following property: $q(\mathcal{D}'') = \mathit{true}$ iff h can be extended to a solution in $p(\mathcal{D})$.

In the second step, we make sure that h can be combined with some mapping on the existentially quantified variables in T', such that no extension to further free variables is possible. To this end, we compute analogous relations R_t also for each node $t \in T'' \setminus T'$. These relations are filled in a bottom-up manner from the leaves of T'' until the leaves of T' are reached. But now we store the additional information for every tuple in R_t indicating if the instantiation of the variables $\bar{y}$ necessarily leads to an extension to some free variable occurring below T''. By deleting from $\mathcal{D}''$ all atoms $R_t(g(\bar{y}))$ with this property, we guarantee that condition $q(\mathcal{D}') = \mathit{true}$ implies that $h \in p(\mathcal{D})$. □

PROOF OF THEOREM 8. Observe that for deciding if h can be extended to some answer $h' \in p(\mathcal{D})$, it suffices to identify some homomorphism $\hat{h}$ from p to $\mathcal{D}$ s.t. $\hat{h}_{\bar{z}} = h$ (where $\bar{z}$ are the variables on which h is defined), i.e., $\hat{h}$ does not need not to be maximal. The problem can thus be solved by (1) identifying the minimal subtree T' of T that contains (at least) all variables from $\bar{z}$; and (2) deciding if $\hat{q}_{T'}(\mathcal{D}) \neq \emptyset$ where $\hat{q}_{T'}$ is derived from $q_{T'}$ by replacing each variable $z \in \bar{z}$ with $h(z)$. Clearly, $\hat{q}_{T'} \in \mathsf{TW}(k)$ (or $\mathsf{HW}(k)$, respectively). Step (1) can be done in LOGSPACE [9], while step (2) fits into LOGCFL [13]. □

PROOF SKETCH OF PROPOSITION 3. We only sketch the lower bound and concentrate on the case $k = 1$. We use a reduction from 3-colorability. Assume $G = (V, E)$ is an undirected graph such that $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$. Let $\mathcal{D} = \{c(1,1), c(2,2), c(3,3)\}$ be a database. We define a WDPT $p = (T, \lambda, \bar{x})$ such that:

- T consists of a root r with children n_j^1, n_j^2 and n_j^3, for each $1 \leq j \leq m$.
- $\lambda(r) = \{c(u_i, u_i) \mid 1 \leq i \leq n\} \cup \{c(x, x)\}$, where x and the u_i's are variables.

- $\lambda(n_j^k) = \{c(u_{j_1}, k), c(u_{j_2}, k), c(x_j^k, x_j^k)\}$, for each $1 \leq j \leq m$ such that $e_j = \{v_{j_1}, v_{j_2}\}$ and $1 \leq k \leq 3$, where the x_j^k's are variables.
- The free variables in $\bar{x}$ are x and all variables of the form x_j^k, for $1 \leq j \leq m$ and $1 \leq k \leq 3$.

We also define a partial mapping $h : \mathbf{X} \to \mathbf{U}$ that satisfies $h(x) = 1$ and is undefined elsewhere.

Clearly, $\mathcal{D}$, p and h can be constructed in polynomial time from G. Furthermore, p belongs to g-$\mathsf{TW}(1)$ and g-$\mathsf{HW}(1)$. We claim that G is 3-colorable iff $h \in p(\mathcal{D})$. This follows directly from the following observation. For every mapping λ from the u_i's to $\{1, 2, 3\}$, it is the case that $h \cup \lambda$ is a maximal homomorphism from p to $\mathcal{D}$ iff λ is a valid 3-coloring of G. The reason why the latter holds is simple. Assume first that λ does not encode a valid 3-coloring of G. Then for some $e_j = \{v_{j_1}, v_{j_2}\}$ it is the case that $\lambda(u_{j_1}) = \lambda(u_{j_2})$. Assume without loss of generality that $\lambda(u_{j_1}) = \lambda(u_{j_2}) = 1$ (the other two cases are analogous). Then $h \cup \lambda \cup h'$ is a homomorphism from p to $\mathcal{D}$, where h' maps x_j^1 to 1. On the other hand, if λ is a valid 3-coloring then clearly no such extension exists. □

A.2 Proof (sketches) for Section 5

Proof of Lemma 1. Let $p = (T, \lambda, \bar{x})$ and $p' = (T_1, \lambda_1, \bar{x}_1)$ be WDPTs such that $p' \sqsubseteq p$ and $p' \in \mathsf{WB}(k)$. We can transform p' into a WDPT $p'' = (T_2, \lambda_2, \bar{x}_1)$ with the desired properties:

First, we restrict the number of nodes in T_1. To this end, we determine the set N of those nodes in T_1 which introduce at least one free variable, i.e., a variable from $\bar{x}_1$ that occurs in this node but not in its parent. Then we delete all nodes that are not on a path from the root to some node in N. Moreover, we may merge every node n with its child node n' if n contains no free variable and n' is the only child of n. We thus end up with a tree T_2 whose number of nodes is linearly bounded in the size of p. Note that it is precisely this merging of nodes where the closure under taking subgraphs of the class $\mathsf{WB}(k)$ is needed.

We then restrict the number of atoms in the labeling λ_1. Recall from [17] that the subsumption test $p' \sqsubseteq p$ requires the existence of certain homomorphisms from certain subtrees of p to subtrees of p'. The number of homomorphisms needed corresponds to the number of subtrees of T_2. Labelling λ_2 is then essentially obtained from λ_1 by deleting all atoms that do not occur in the image of any of these homomorphisms. Here we get an exponential blow-up due to the number of subtrees (and, hence, of homomorphisms) that have to be considered. □

Proof of Corollary 2. From Theorem 13, there is a NExptime$^{\text{NP}}$ algorithm that, given a WDPT p in $\mathcal{M}(\mathsf{WB}(k))$, constructs an exponential size p' in $\mathsf{WB}(k)$ such that $p \equiv_s p'$. Therefore, in order to check whether a partial mapping $h : \mathbf{X} \to \mathbf{U}$ is a partial answer to p over $\mathcal{D}$, for p a WDPT in $\mathcal{M}(\mathsf{WB}(k))$ and $\mathcal{D}$ a database, we can construct p' using the previous algorithm and then check whether h is a partial answer to p' over $\mathcal{D}$ (since this is equivalent with the fact that h is a partial answer to p over $\mathcal{D}$). The latter can be solved in polynomial time in the size of $\mathcal{D}$ and p' from Theorem 9. We conclude that Partial-Eval($\mathcal{M}(\mathsf{WB}(k))$) can be solved in time $O(f(|p|) + |\mathcal{D}|^c \cdot 2^{t(|p|)})$, where $f : \mathbb{N} \to \mathbb{N}$ is a double-exponential function, $t : \mathbb{N} \to \mathbb{N}$ is a polynomial, and $c \geq 1$ is a constant. A similar argument shows that Max-Eval($\mathcal{M}(\mathsf{WB}(k))$) is fixed-parameter tractable. □

Proof sketch of Theorem 14. It follows from the proof of Lemma 1 that there is a polynomial $t : \mathbb{N} \to \mathbb{N}$, such that the $\mathsf{WB}(k)$-approximations of a WDPT p are precisely the maximal elements (with respect to $\sqsubseteq$) of the set of WDPTs in $\mathsf{WB}(k)$ that are subsumed by p and whose size is at most $2^{t(|p|)}$. It follows from [4] that this set is nonempty, and therefore it contains at least one maximal element. Furthermore, each such maximal element can be computed in double-exponential time from p. □

A.3 Proof (sketches) for Section 6

Proof sketch of Proposition 9. For the right-to-left direction observe that ϕ_{cq} is $\equiv_s$-equivalent to a union of CQs in $\mathcal{C}(k)$, i.e., a union of single-node WDPTs in $\mathsf{WB}(k)$.

For the left-to-right direction assume that ϕ is $\equiv_s$-equivalent to a UWDPT ϕ^* in $\mathsf{UWB}(k)$. Then $\phi \equiv_s \phi^* \equiv_s \phi_{\text{cq}} \equiv_s \phi^*_{\text{cq}}$. But for unions of CQs we have that $\equiv_s$ is the same as $\equiv$, and hence $\phi_{\text{cq}} \equiv \phi^*_{\text{cq}}$. Moreover, since each WDPT in ϕ^* is in $\mathsf{WB}(k)$, it follows that ϕ^*_{cq} is indeed a union of CQs in $\mathcal{C}(k)$. □

Proof of Theorem 17. Let ϕ^r_{cq} be the union of CQs that is obtained by removing from ϕ_{cq} every CQ q that is contained in another CQ q' in ϕ_{cq}. By Proposition 9, query ϕ is in $\mathcal{M}(\mathsf{UWB}(k))$ iff ϕ_{cq} is equivalent to a union of CQs in $\mathcal{C}(k)$. It is easy to prove that this is the case iff each CQ in ϕ^r_{cq} is equivalent to a CQ in $\mathcal{C}(k)$ [5].

This gives us the following non-deterministic algorithm to check if $\phi \notin \mathcal{M}(\mathsf{UWB}(k))$:

Guess a CQ $q \in \phi_{\text{cq}}$ (all of them are of polynomial size);
check that (1) $q \not\sqsubseteq q'$ for every other CQ q' in ϕ_{cq}, and
(2) q is not equivalent to a CQ in $\mathcal{C}(k)$.

Clearly, (1) can be checked in coNP, and the same holds for (2) in case of $\mathcal{C}(k) = \mathsf{TW}(k)$ (see, e.g., [4]). For $\mathcal{C}(k) = \mathsf{HW}'(k)$, step (2) can be checked in Π_2^P, thus giving us the desired Π_2^P- and Π_3^P-algorithms, respectively.

The proof of the second part of the theorem is similar. □

Proof sketch of Theorem 18. By $\phi \equiv_s \phi_{\text{cq}}$, we can apply results on approximations of unions of CQs from [4]. Let $\phi_{\text{cq}-\text{app}}$ denote the union of all $\mathcal{C}(k)$-approximations of the CQs in ϕ_{cq}. A crucial result in [4] is that the $\mathcal{C}(k)$-approximation of a union of CQs can be obtained as the union of the approximations. Moreover, it follows from further results in [4] that $\phi_{\text{cq}-\text{app}}$ is nonempty and contains at most a single-exponential number of CQs. Furthermore, each CQ in $\phi_{\text{cq}-\text{app}}$ can be assumed to be of polynomial size. Hence, $\phi_{\text{cq}-\text{app}}$ is the desired UWDPT ϕ'. □

Proof of Proposition 10. Hardness follows from the Π_2^P-hardness of Subsumption, which holds as long as the restriction to $\mathsf{WB}(k)$ only applies to the WDPT on the left-hand side. Now look at the membership: Given two UWDPTs ϕ', ϕ, to test if ϕ' is a $\mathsf{UWB}(k)$-approximation of ϕ, we first check that $\phi' \sqsubseteq \phi$ holds, which can be done in Π_2^P [19]. If this is the case, the proof of Theorem 18 tells us that ϕ' is a $\mathsf{UWB}(k)$-approximation of ϕ iff $\phi_{\text{cq}-\text{app}} \sqsubseteq \phi'$, where $\phi_{\text{cq}-\text{app}}$ again denotes the union of all $\mathcal{C}(k)$-approximations of the CQs in ϕ_{cq}. Checking if the latter is the case can be done in Π_2^P for $\mathcal{C}(k) = \mathsf{TW}(k)$ and in Π_3^P for $\mathcal{C}(k) = \mathsf{HW}(k)$. □