The Tractability Frontier of Well-designed SPARQL Queries[*]

Miguel Romero
University of Oxford, UK
miguel.romero@cs.ox.ac.uk

ABSTRACT

We study the complexity of query evaluation of SPARQL queries. We focus on the fundamental fragment of well-designed SPARQL restricted to the AND, OPTIONAL and UNION operators. Our main result is a structural characterisation of the classes of well-designed queries that can be evaluated in polynomial time. In particular, we introduce a new notion of width called *domination width*, which relies on the well-known notion of treewidth. We show that, under some complexity theoretic assumptions, the classes of well-designed queries that can be evaluated in polynomial time are precisely those of bounded domination width.

CCS CONCEPTS

• Information systems → Resource Description Framework (RDF); **• Theory of computation → Database query processing and optimization (theory)**;

KEYWORDS

well-designed SPARQL, pattern trees, treewidth, domination width, evaluation, polynomial time

ACM Reference Format:
Miguel Romero. 2018. The Tractability Frontier of Well-designed SPARQL Queries. In *PODS'18: 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, June 10–15, 2018, Houston, TX, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3196959.3196973

1 INTRODUCTION

The *Resource Description Framework* (RDF) [20] is the W3C standard for representing linked data on the Web. In this model, data is represented as *RDF graphs*, which consist of collections of triples of *internationalised resource identifiers* (IRIs). Intuitively, such a triple (s, p, o) represents the fact that a *subject* s is connected to an *object* o via a *predicate* p.

SPARQL [26] is the standard query language for RDF graphs. In a seminal paper, Pérez et al. [23] (see also [22]) gave a clean formalisation of the language, which laid the foundations for its theoretical study. Since then, a lot of work has been done in different aspects of the language such as query evaluation [3, 4, 15, 16, 19], optimisation [14, 17, 24], and expressive power [2, 11, 15, 25, 30], to name a few.

As shown in [23], it is PSPACE-complete to evaluate SPARQL queries. This motivated the introduction of a natural fragment of SPARQL called the *well-designed* fragment, whose evaluation problem is coNP-complete [23]. More formally, the *evaluation problem* wdEVAL for well-designed SPARQL is to decide, given a well-designed query P, and RDF graph G and a mapping μ, whether μ belongs to the answer $[\![P]\!]_G$ of P over G. By now the well-designed fragment is central in the study of SPARQL and a lot of efforts has been done by the theory community to understand fundamental aspects of this fragment (see e.g. [4, 11, 14–17, 23, 24]). In this paper, we focus on the core fragment of well-designed SPARQL restricted to the AND, OPTIONAL and UNION operators, as defined in [23].

Despite its importance, several basic questions remain open for well-designed SPARQL. As first observed in [17], while the problem wdEVAL is coNP-complete, it becomes *tractable*, i.e. polynomial-time solvable, for restricted classes of well-designed queries. Indeed, it was shown that wdEVAL is in PTIME for every class C of queries satisfying a certain *local tractability* condition [23]. We emphasise that the above-mentioned result is briefly discussed in [23] as the focus of the authors is on the static analysis and optimisation of queries rather than complexity of evaluation. Subsequent works [4, 16] have studied the complexity of evaluation in more depth but the focus has been mainly on the fragment of SPARQL including the SELECT operator (i.e., *projection*). In particular, the following fundamental question regarding the core well-designed fragment remains open: *which classes C of well-designed SPARQL can be evaluated in polynomial time?*

Our main contribution is a complete answer to the question posed above. In particular, we introduce a new width measure for well-designed queries called *domination width*, which is based on the well-known notion of *treewidth* (see Section 3 for precise definitions). For a class C of well-designed queries, let us denote by wdEVAL(C) the evaluation problem wdEVAL restricted to the class C. Also, we say that a class C of well-designed queries has *bounded* domination width if there is an universal constant $k \geq 1$ such that the domination width of every query in C is at most k. Then, our main technical result is as follows (Theorem 4.2). *Assume that FPT $\neq$ W[1]. Then, for every recursively enumerable class C of well-designed queries, the problem* wdEVAL(C) *is in PTIME if and only if C has bounded domination width.* The assumption FPT $\neq$ W[1] is a widely believed assumption from parameterised complexity (see Section 4 for precise definitions). As we observe in Section 3, one can remove the assumption of C being recursively enumerable by considering a stronger assumption than FPT $\neq$ W[1] considering non-uniform complexity classes.

[*]This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532). The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.

Our result builds on the classical result by Dalmau et al. [6] and Grohe [9] showing that a recursively enumerable class of *conjunctive queries* (CQs) over schemas of *bounded arity* is tractable if and only if the *cores* of the CQs in C have bounded treewidth. (Recall that a CQ is a first-order query using only conjunctions and existential quantification.)

For the tractability part of our result, we exploit, as in [6], the so-called *existential pebble game* introduced in [12] (see also [6]). This game provides a polynomial-time relaxation for the problem of checking the existence of homomorphisms, which is a well-known NP-complete problem (see e.g. [5]). Using the existential pebble game, we define a natural relaxation of the standard algorithm from [17] (see also [24]) for evaluating well-designed queries. Then we show that this relaxation correctly solves instances of bounded domination width (Theorem 3.10).

For the hardness part, we follow a similar strategy as in [9]. The two main ingredients in our proof is an adaptation of the main construction of [9] to handle *distinguished elements* or *constants* (Lemma 4.4) and an elementary property of well-designed queries of large domination width (Lemma 4.5).

Finally, we emphasise that our classes of bounded domination width significantly extend the classes that are locally tractable [17], which, as we mentioned above, are the most general tractable restrictions known so far. This is even true in the case of UNION-free well-designed queries. As we discuss in Section 3.2, the notion of domination width for UNION-free queries can be simplified and coincides with a width measure called *branch treewidth*. Bounding this simpler width measure still *strictly* generalises local tractability.

Organisation. We present the basic definitions in Section 2. In Section 3, we introduce the measure of domination width and present our main tractability result. The main hardness result is presented in Section 4. We conclude with some final remarks in Section 5.

2 PRELIMINARIES

RDF Graphs. Let $\mathbf{I}$ be a countable infinite set of IRIs. An *RDF triple* is a tuple in $\mathbf{I} \times \mathbf{I} \times \mathbf{I}$ and an *RDF graph* is a finite set of RDF triples. In this paper, we assume that no blank nodes appear in RDF graphs, i.e., we focus on *ground* RDF graphs.

SPARQL Syntax. SPARQL [26] is the standard query language for RDF. We rely on the formalisation proposed in [23]. We focus on the core fragment of the language given by the operators AND, OPTIONAL (OPT for short), and UNION.[1] Let $\mathbf{V} = \{?x, ?y, \dots\}$ be a countable infinite set of *variables*, disjoint from $\mathbf{I}$. A SPARQL *triple pattern* (or *triple pattern* for short) is a tuple in $(\mathbf{I}\cup\mathbf{V})\times(\mathbf{I}\cup\mathbf{V})\times(\mathbf{I}\cup\mathbf{V})$. The set of variables from $\mathbf{V}$ appearing in a triple pattern t is denoted by $\text{vars}(t)$. Note that an RDF triple is simply a SPARQL triple pattern t with $\text{vars}(t) = \emptyset$. A SPARQL *graph pattern* (or *graph pattern* for short) is recursively defined as follows:

(1) a triple pattern is a graph pattern, and
(2) if P_1 and P_2 are graph patterns, then $P_1 * P_2$ is also a graph pattern, for $* \in \{\text{AND}, \text{OPT}, \text{UNION}\}$.

[1]Additional operators include FILTER and SELECT. We briefly discuss these operators in Section 5.

SPARQL Semantics. In order to define the semantics of graph patterns, we follow again the presentation in [23]. A *mapping* μ is a partial function from $\mathbf{V}$ to $\mathbf{I}$. We denote by $\text{dom}(\mu)$ the domain of the mapping μ. Two mappings μ_1 and μ_2 are *compatible* if $\mu_1(?x) = \mu_2(?x)$, for all $?x \in \text{dom}(\mu_1) \cap \text{dom}(\mu_2)$. If μ_1 and μ_2 are compatible mappings then $\mu_1 \cup \mu_2$ denotes the mapping with domain $\text{dom}(\mu_1) \cup \text{dom}(\mu_2)$ such that $\mu_1 \cup \mu_2(?x) = \mu_1(?x)$, for all $?x \in \text{dom}(\mu_1)$, and $\mu_1 \cup \mu_2(?x) = \mu_2(?x)$, for all $?x \in \text{dom}(\mu_2)$. For a triple pattern t and a mapping μ such that $\text{vars}(t) \subseteq \text{dom}(\mu)$, we denote by $\mu(t)$ the RDF triple obtained from t by replacing each $?x \in \text{vars}(t)$ by $\mu(?x)$.

For an RDF graph G and a graph pattern P, the *evaluation* $[\![P]\!]_G$ of P over G is a set of mappings defined recursively as follows:

(1) $[\![t]\!]_G = \{\mu \mid \text{dom}(\mu) = \text{vars}(t) \text{ and } \mu(t) \in G\}$, if t is a triple pattern.
(2) $[\![P_1 \text{ AND } P_2]\!]_G = \{\mu_1 \cup \mu_2 \mid \mu_1 \in [\![P_1]\!]_G, \mu_2 \in [\![P_2]\!]_G$ and μ_1, μ_2 are compatible$\}$.
(3) $[\![P_1 \text{ OPT } P_2]\!]_G = [\![P_1 \text{ AND } P_2]\!]_G \cup \{\mu_1 \mid \mu_1 \in [\![P_1]\!]_G$ and there is no $\mu_2 \in [\![P_2]\!]_G$ compatible with $\mu_1\}$.
(4) $[\![P_1 \text{ UNION } P_2]\!]_G = [\![P_1]\!]_G \cup [\![P_2]\!]_G$.

Well-designed SPARQL. A central class of SPARQL graph patterns identified in [23], and also the focus of this paper, is the class of well-designed graph patterns. We say that a graph pattern is UNION-*free* if it only uses the operators AND and OPT. A UNION-free graph pattern P is *well-designed* if for every subpattern $P' = (P_1 \text{ OPT } P_2)$ of P, it is the case that every variable $?x$ ocurring in P_2 but not in P_1, does *not* occur outside P' in P. A SPARQL graph pattern P is *well-designed* if it is of the form $P = P_1 \text{ UNION} \cdots \text{UNION } P_m$, where each P_i is a UNION-free well-designed graph pattern.[2]

Example 2.1. Consider the following graph patterns:

$$P_1 = ((?x, p, ?y) \text{ OPT } (?z, q, ?x)) \text{ OPT } ((?y, r, ?o_1) \text{ AND } (?o_1, r, ?o_2)),$$
$$P_2 = ((?x, p, ?y) \text{ OPT } (?z, q, ?x)) \text{ OPT } ((?y, r, ?z) \text{ AND } (?z, r, ?o_2)).$$

Note that P_1 is well-designed, while P_2 is not. Indeed, in the subpattern $P_2' = ((?x, p, ?y) \text{ OPT } (?z, q, ?x))$ of P_2, the variable $?z$ appears in $(?z, q, ?x)$ and not in $(?x, p, ?y)$ but *does* occur outside P_2' in P_2.

Well-designed patterns have good properties in terms of query evaluation. More precisely, let wdEVAL be the problem of deciding, given a well-designed graph pattern P, an RDF graph G and a mapping μ, whether $\mu \in [\![P]\!]_G$. It was shown in [23] that wdEVAL is coNP-complete, while the problem is PSPACE-complete for arbitrary SPARQL graph patterns.

2.1 Pattern trees and pattern forests

Besides alleviating the cost of evaluation, another key property of UNION-free well-designed graph patterns is that they can be written in the so-called OPT-normal form [23]. In turn, patterns in OPT-normal form admit a natural tree representation, known as *pattern trees* [17]. Intuitively, a pattern tree is a rooted tree where each node represents a well-designed pattern using only AND operators, while its tree structure represents the nesting of OPT operators. Consequently, a well-designed graph pattern $P = P_1$ UNION $\cdots$

[2]This top-level use of the UNION operator is known as UNION-normal form [23]. Note that we are implicitly using the fact that UNION is associative.

UNION P_m can be represented as a *pattern forest*[3][24], i.e., a set of pattern trees $\{\mathcal{T}_1, \dots, \mathcal{T}_m\}$, where $\mathcal{T}_i$ is the pattern tree representation of P_i. Pattern trees/forests are useful for understanding how to evaluate and optimise well-designed patterns, and have been used extensively as a basic tool in the study of well-designed SPARQL (see e.g. [4, 11, 14, 16, 17, 24]). As we show in this work, pattern forests are also fundamental to understand tractable evaluation of well-designed SPARQL: by imposing restrictions on the pattern forest representation, we can identify and *characterise* the tractable classes of well-designed graph patterns.

T-graphs and homomorphisms. A *triple pattern graph* (or *t-graph* for short) is a finite set S of triple patterns. We denote by $\text{vars}(S)$ the set of variables from $\mathbf{V}$ appearing in the t-graph S. Note that an RDF graph is simply a t-graph S with $\text{vars}(S) = \emptyset$. Let t be a triple pattern and h be a partial function from $\mathbf{V}$ to $\mathbf{I} \cup \mathbf{V}$ such that $\text{vars}(t) \subseteq \text{dom}(h)$. We define $h(t)$ to be the triple pattern obtained from t by replacing each $?x \in \text{vars}(t)$ by $h(?x)$. For two t-graphs S and S', we say that a partial function h from $\mathbf{V}$ to $\mathbf{I} \cup \mathbf{V}$ is a *homomorphism* from S to S' if $\text{dom}(h) = \text{vars}(S)$ and for every $t \in S$, it is the case that $h(t) \in S'$.

Basics of pattern trees and forests. For an undirected graph H, we denote by $V(H)$ its set of nodes. A *well-designed pattern tree* (or wdPT for short) is a triple $\mathcal{T} = (T, r, \lambda)$ such that

(1) T is a tree rooted at a node $r \in V(T)$,
(2) λ is a function that maps each node $n \in V(T)$ to a t-graph, and
(3) the set $\{n \in V(T) \mid ?x \in \text{vars}(\lambda(n))\}$ induces a connected subgraph of T, for every $?x \in \mathbf{V}$.

Let $\mathcal{T} = (T, r, \lambda)$ be a wdPT. A wdPT $\mathcal{T}' = (T', r', \lambda')$ is a *subtree* of $\mathcal{T}$ if (i) T' is a subtree of T, (ii) $r' = r$, and $\lambda'(n) = \lambda(n)$, for all $n \in V(T')$. Note that any subtree of $\mathcal{T}$ contains the original root r. A *child* of the subtree $\mathcal{T}'$ is a node $n \in V(T) \setminus V(T')$ such that $n' \in V(T')$, where n' is the parent of n in T.

For convenience, we fix two functions $\text{pat}(\cdot)$ and $\text{vars}(\cdot)$ as follows. Let $\mathcal{T} = (T, r, \lambda)$ be a wdPT. We define $\text{pat}(n) := \lambda(n)$, for every $n \in V(T)$ and $\text{pat}(\mathcal{T}) := \bigcup_{n \in V(T)} \text{pat}(n)$. Note that $\text{pat}(n)$ and $\text{pat}(\mathcal{T})$ are t-graphs. We let $\text{vars}(n) := \text{vars}(\text{pat}(n))$, for $n \in V(T)$ and $\text{vars}(\mathcal{T}) := \text{vars}(\text{pat}(\mathcal{T}))$.

A *well-designed pattern forest* (wdPF for short) is a finite set $\mathcal{F} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ of well-designed pattern trees.

In [17], it was shown that every wdPT can be translated efficiently into an equivalent wdPT in the so-called NR normal form. A wdPT $\mathcal{T} = (T, r, \lambda)$ is in *NR normal form* if for every node $n \in V(T)$ with parent n' in T, it holds that $\text{vars}(n) \setminus \text{vars}(n') \neq \emptyset$. In this paper, we assume that all wdPTs are in NR normal form.

Well-designed SPARQL and wdPFs. As in the case of SPARQL graph patterns, we denote by $[\![\mathcal{T}]\!]_G$ (resp., $[\![\mathcal{F}]\!]_G$) the evaluation of a wdPT $\mathcal{T}$ (resp., wdPF $\mathcal{F}$) over an RDF graph G. In [17], for a wdPT $\mathcal{T}$, the set of mappings $[\![\mathcal{T}]\!]_G$ is defined via a translation to well-designed graph patterns. However, if $\mathcal{T}$ is in NR-normal form, then $[\![\mathcal{T}]\!]_G$ admits a simple characterisation stated in Lemma 2.2 below. In this paper, we adopt this characterisation as the semantics of wdPTs.

Lemma 2.2 ([17, 24]). *Let $\mathcal{T}$ be a wdPT in NR normal form, G an RDF graph and μ a mapping. Then $\mu \in [\![\mathcal{T}]\!]_G$ iff there exists a subtree $\mathcal{T}'$ of $\mathcal{T}$ such that*

(1) *μ is a homomorphism from $\text{pat}(\mathcal{T}')$ to G.*
(2) *there is no child n of $\mathcal{T}'$ and homomorphism ν from $\text{pat}(n)$ to G compatible with μ.*

For a wdPF $\mathcal{F} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ and an RDF graph G, we define $[\![\mathcal{F}]\!]_G = [\![\mathcal{T}_1]\!]_G \cup \dots \cup [\![\mathcal{T}_m]\!]_G$.

As shown in [17], every UNION-free well-designed graph pattern P can be translated in polynomial time into an equivalent wdPT $\mathcal{T}$, i.e., a wdPT such that $[\![\mathcal{T}]\!]_G = [\![P]\!]_G$, for all RDF graphs G. Consequently and as observed in [24], every well-designed graph pattern P can be translated in polynomial time into an equivalent wdPF $\mathcal{F}$. *Throughout the paper, we fix a polynomial-time computable function* **wdpf** *that maps each well-designed graph pattern to an equivalent wdPF.*

Example 2.3. Recall P_1 from Example 2.1 and consider the following well-designed graph pattern:

$$P = P_1 \text{ UNION } ((?x, p, ?y) \text{ OPT } ((?z, q, ?x) \text{ AND } (?w, q, ?z))).$$

We have that $\mathbf{wdpf}(P) = \{\mathcal{T}_1, \mathcal{T}_2\}$, where $\mathcal{T}_1$ and $\mathcal{T}_2$ are the wdPTs depicted in Figure 2, for $k = 2$ and $K_2(?o_1, ?o_2) = \{(?o_1, r, ?o_2)\}$.

2.2 Restrictions of the evaluation problem

Recall that wdEVAL denotes the problem of deciding, given a well-designed graph pattern P, an RDF graph G and a mapping μ, whether $\mu \in [\![P]\!]_G$. In this paper, we study restrictions of wdEVAL given by different classes C of admissible patterns. Formally, for a class C of well-designed graph patterns, we define the problem wdEVAL(C) as follows:

wdEVAL(C)
Input: a well-designed graph pattern $P \in C$, an RDF graph G and a mapping μ.
Question: does $\mu \in [\![P]\!]_G$ hold?

Note that wdEVAL(C) is a *promise* problem, as we are given the promise that $P \in C$. This allows us to analyse the complexity of evaluating patterns in C *independently* of the cost of checking membership in C.

3 A NEW TRACTABILITY CONDITION

In this section, we introduce the notion of domination width of a well-designed graph pattern and show our main tractability result: wdEVAL(C) is in PTIME, for classes C of graph patterns of bounded domination width. Before doing so, we need to introduce some terminology.

A *generalised t-graph* is a pair (S, X), where S is a t-graph and $X \subseteq \text{vars}(S)$. Consider two generalised t-graphs of the form (S, X) and (S', X). A *homomorphism* from (S, X) to (S', X) is a homomorphism h from S to S' such that $h(?x) = ?x$, for all $?x \in X$. We write $(S, X) \to (S', X)$ whenever there is a homomorphism from (S, X) to (S', X); otherwise, we write $(S, X) \not\to (S', X)$. Note that the relation $\to$

[3]In this paper, we work with a particular type of patterns trees/forests, namely *well-designed pattern trees/forests.* For simplicity, sometimes we abuse notation and use the terms patterns trees/forests and well-designed pattern trees/forests interchangeably.

is transitive, i.e., $(S,X) \to (S',X)$ and $(S',X) \to (S'',X)$ implies $(S,X) \to (S'',X)$.

Let (S,X) be a generalised t-graph, G be an RDF graph and μ be a mapping with $\mathrm{dom}(\mu) = X$. We write $(S,X) \to^{\mu} G$ if there is a homomorphism h from S to G such that $h(?x) = \mu(?x)$, for all $?x \in X$. Notice that $\to$ composes with $\to^{\mu}$, i.e., $(S,X) \to (S',X)$ and $(S',X) \to^{\mu} G$ implies $(S,X) \to^{\mu} G$.

Below we state several notions and properties for generalised t-graphs. We emphasise that all these properties are well-known for *conjunctive queries* (CQs) and *relational structures* and can be applied in our case as there is a strong correspondence between generalised t-graphs and CQs. Indeed, we can view a generalised t-graph (S,X) as a CQ $q_{(S,X)}$ over a relational schema containing a single ternary relation, where the variables are $\mathrm{vars}(S)$, the *free variables* are X, and the IRIs appearing in S correspond to *constants* in $q_{(S,X)}$. However, for convenience and consistency with RDF and SPARQL terminology, we shall work directly with generalised t-graphs throughout the paper.

Cores. Let (S,X) and (S',X) be two generalised t-graphs. We say that (S',X) is a *subgraph* of (S,X) if $S' \subseteq S$, and a *proper* subgraph if $S' \subseteq S$ but $S \nsubseteq S'$. A generalised t-graph (S,X) is a *core* if there is no homomorphism from (S,X) to one of its proper subgraphs (S',X). We say that (S',X) is a *core* of (S,X) if (S',X) is a core itself, $(S,X) \to (S',X)$ and $(S',X) \to (S,X)$. As stated below, every generalised t-graph (S,X) has a unique core (up to renaming of variables), and hence, we can speak of *the* core of a generalised t-graph.

Proposition 3.1 (see e.g. [1, 10]). *Every generalised t-graph (S,X) has a unique core (S',X) (up to renaming of variables).*

Treewidth. The notion of treewidth is a well-known measure of the tree-likeness of an undirected graph (see e.g. [7]). For instance, trees have treewidth 1, cycles treewidth 2 and K_k, the clique of size k, treewidth $k-1$. Let H be an undirected graph. A *tree decomposition* of H is a pair (F,β) where F is a tree and β is a function that maps each node $s \in V(F)$ to a subset of $V(H)$ such that

(1) for every $u \in V(H)$, the set $\{s \in V(F) \mid u \in \beta(s)\}$ induces a connected subgraph of F, and
(2) for every edge $\{u,v\} \in E(H)$, there is a node $s \in V(F)$ with $\{u,v\} \subseteq \beta(s)$.

The *width* of the decomposition (F,β) is $\max\{|\beta(s)| \mid s \in V(F)\} - 1$. The *treewidth* $\mathrm{tw}(H)$ of the graph H is the minimum width over all its tree decompositions.

Let (S,X) be a generalised t-graph. The *Gaifman graph* $G(S,X)$ of (S,X) is the undirected graph whose vertex set is $\mathrm{vars}(S) \setminus X$ and whose edge set contains the pairs $\{?x,?y\}$ such that $?x \neq ?y$ and $\{?x,?y\} \subseteq \mathrm{vars}(t)$, for some triple pattern $t \in S$. We define the *treewidth* of (S,X) to be $\mathrm{tw}(S,X) := \mathrm{tw}(G(S,X))$. If $G(S,X)$ has no vertices, i.e., $\mathrm{vars}(S) \setminus X = \emptyset$, or $G(S,X)$ has no edges, we let $\mathrm{tw}(S,X) = \mathrm{tw}(G(S,X)) := 1$.

For a generalised t-graph (S,X), we let $\mathrm{ctw}(S,X) := \mathrm{tw}(S',X)$, where (S',X) is the core of (S,X).

Example 3.2. Let $X = \{?x,?y,?z\}$ and consider the generalised t-graphs (S,X) and (S',X) depicted in Figure 1, where $k \geq 2$ and $K_k(?o_1,\ldots,?o_k)$ is the t-graph given by the set

$$K_k(?o_1,\ldots,?o_k) := \{(?o_i,r,?o_j) \mid i,j \in \{1,\ldots,k\} \text{ with } i<j\}.$$

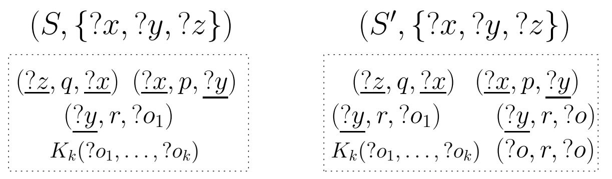
$(S,\{?x,?y,?z\})$
$(\underline{?z},q,\underline{?x})$ $(\underline{?x},p,\underline{?y})$
$(\underline{?y},r,?o_1)$
$K_k(?o_1,\ldots,?o_k)$
$(S',\{?x,?y,?z\})$
$(\underline{?z},q,\underline{?x})$ $(\underline{?x},p,\underline{?y})$
$(\underline{?y},r,?o_1)$ $(\underline{?y},r,?o)$
$K_k(?o_1,\ldots,?o_k)$ $(?o,r,?o)$

Figure 1: The generalised t-graphs from Example 3.2. We assume that $k \geq 2$ and $K_k(?o_1,\ldots,?o_k) = \{(?o_i,r,?o_j) \mid i,j \in \{1,\ldots,k\}$ with $i<j\}$. Note that the distinguished variables are underlined.

Observe that (S,X) is a core and hence $\mathrm{ctw}(S,X) = k-1$, as its Gaifman graph is the clique of size k. On the other hand, the core of (S',X) is (C',X), where

$$C' = \{(?z,q,?x),(?x,p,?y),(?y,r,?o),(?o,r,?o)\}.$$

Hence, $\mathrm{ctw}(S',X) = 1$ while $\mathrm{tw}(S',X) = k-1$.

Existential k-pebble game. The *existential k-pebble game* was introduced by Kolaitis and Vardi [12] to analyse the expressive power of certain *Datalog programs*. While the original definition deals with relational structures, here we focus on the natural adaptation to the context of generalised t-graphs and RDF graphs.

Let $k \geq 2$. The existential k-pebble game is played by the *Spoiler* and the *Duplicator* on a generalised t-graph (S,X), an RDF graph G and a mapping μ with $\mathrm{dom}(\mu) = X$. During the game, the Spoiler only picks elements from $\mathrm{vars}(S) \setminus X$, while the Duplicator picks elements from $\mathrm{dom}(G)$, where $\mathrm{dom}(G) \subseteq \mathbf{I}$ is the set of IRIs appearing in G. In the first round, the Spoiler places pebbles on (not necessarily distinct) elements $?x_1,\ldots,?x_k \in \mathrm{vars}(S) \setminus X$, and the Duplicator responds by placing pebbles on elements $a_1,\ldots,a_k \in \mathrm{dom}(G)$. On any further round, the Spoiler removes a pebble and places it on another element $?x \in \mathrm{vars}(S) \setminus X$. The Duplicator responds by moving the corresponding pebble to an element $a \in \mathrm{dom}(G)$. If after a particular round, the elements covered by the pebbles are $?x_1,\ldots,?x_k$ and $a_1,\ldots,a_k$ for the Spoiler and the Duplicator, respectively, then the *configuration* of the game is $\perp$ if $?x_i = ?x_j$ and $a_i \neq a_j$, for some $i,j \in \{1,\ldots,k\}$ with $i \neq j$; otherwise, it is the mapping $\mu \cup \nu$, where $\mathrm{dom}(\nu) = \{?x_1,\ldots,?x_k\}$ and $\nu(?x_i) = a_i$, for every $i \in \{1,\ldots,k\}$ (note that $\mathrm{dom}(\mu) \cap \mathrm{dom}(\nu) = \emptyset$).

The Duplicator wins the game if he has a *winning strategy*, that is, he can indefinitely continue playing the game in such a way that the configuration at the end of each round is a mapping $\mu \cup \nu$ that is a *partial homomorphism*, i.e., for every triple pattern $t \in S$ with $\mathrm{vars}(t) \subseteq \mathrm{dom}(\mu \cup \nu)$, it is the case that $\mu \cup \nu(t) \in G$. If the Duplicator can win the existential k-pebble game on (S,X), G and μ, then we write $(S,X) \to_k^{\mu} G$.

Note that if $\mathrm{vars}(S) \setminus X = \emptyset$, then for every $k \geq 2$,

$$(S,X) \to_k^{\mu} G \text{ if and only if } (S,X) \to^{\mu} G, \quad (1)$$

i.e., μ is a homomorphism from S to G. Observe also that for every $k \geq 2$,

$$(S,X) \to^{\mu} G \text{ implies } (S,X) \to_k^{\mu} G. \quad (2)$$

In other words, the relation $\rightarrow_k^\mu$ is a *relaxation* of $\rightarrow^\mu$. As we state below, the relaxation given by $\rightarrow_k^\mu$ has good properties in terms of complexity[4]: while checking the existence of homomorphisms, i.e., $(S, X) \rightarrow^\mu G$ is a well-known NP-complete problem [5], checking $(S, X) \rightarrow_k^\mu G$ can be done in polynomial time, for every fixed $k \geq 2$.

Proposition 3.3 ([12]; see also [6]). *Let $k \geq 2$. For a given generalised t-graph (S, X), an RDF graph G and a mapping μ with $dom(\mu) = X$, checking whether $(S, X) \rightarrow_k^\mu G$ can be done in polynomial time.*

As it turns out, there is a strong connection between existential k-pebble games and the notion of treewidth. In particular, it was shown by Dalmau et al. [6] that the relations $\rightarrow_k$ and $\rightarrow$ coincide for generalised t-graphs (S, X) satisfying $\text{ctw}(S, X) \leq k - 1$[5].

Proposition 3.4 ([6]). *Let $k \geq 2$. Let (S, X) be a generalised t-graph, G be an RDF graph and μ be a mapping with $dom(\mu) = X$. Suppose that $ctw(S, X) \leq k - 1$. Then $(S, X) \rightarrow_k^\mu G$ if and only if $(S, X) \rightarrow^\mu G$.*

We conclude with two basic properties of the existential pebble game that will be useful for us.

Proposition 3.5. *Let $k \geq 2$. Let $(S_1, X), (S_2, X), \ldots, (S_\ell, X)$ be generalised t-graphs ($\ell \geq 2$), G be an RDF graph and μ be a mapping with $dom(\mu) = X$. Then the following hold:*

(1) *if $(S_1, X) \rightarrow (S_2, X)$ and $(S_2, X) \rightarrow_k^\mu G$, then it is the case that $(S_1, X) \rightarrow_k^\mu G$.*
(2) *if $(S_i, X) \rightarrow_k^\mu G$, for all $i \in \{1, \ldots, \ell\}$ and $(vars(S_i) \setminus X) \cap (vars(S_j) \setminus X) = \emptyset$, for all $i, j \in \{1, \ldots, \ell\}$ with $i \neq j$, then $(S_1 \cup \cdots \cup S_\ell, X) \rightarrow_k^\mu G$.*

3.1 Domination width

We start by giving some intuition regarding the notion of domination width. Let P be a well-designed graph pattern, G be an RDF graph and μ be a mapping. Suppose that $\mathbf{wdpf}(P) = \mathcal{F}$ and $\mathcal{F} = \{\mathcal{T}_1, \ldots, \mathcal{T}_m\}$, for $m \geq 1$. The natural algorithm for checking $\mu \notin [\![\mathcal{F}]\!]_G$ is as follows (see e.g. [17, 24]): we simply iterate over all $i \in \{1, \ldots, m\}$ such that μ is a *potential solution* of $\mathcal{T}_i$ over G, i.e., there is a subtree $\mathcal{T}_i'$ of $\mathcal{T}_i$ such that μ is a homomorphism from $\text{pat}(\mathcal{T}_i')$ to G, and we ensure that there is a child n_i of $\mathcal{T}_i'$ where μ can be extended consistently.

The key observation is that we can reinterpret the above-described algorithm as follows. We can choose one of the subtrees $\mathcal{T}_i'$ as above, and associate a collection of generalised t-graphs $\mathbf{GtG}(\mathcal{T}_i')$ of the form $(S, \text{vars}(\mathcal{T}_i'))$, where $S = \text{pat}(\mathcal{T}_i') \cup \bigcup_{j \in I} \text{pat}(n_j)$, where $I \subseteq \{1, \ldots, m\}$ is the set of indices j such that μ is a potential solution of $\mathcal{T}_j$ over G, and n_j is a child of $\mathcal{T}_j'$. To avoid conflicts, for every $j \in I$, the variables from $\text{vars}(n_j)$ that are not in $\text{vars}(\mathcal{T}_i') = \text{dom}(\mu)$, need to be renamed to fresh variables. Therefore, checking $\mu \notin [\![\mathcal{F}]\!]_G$ amounts to checking that there is a homomorphism from some element of $\mathbf{GtG}(\mathcal{T}_i')$ to G, i.e., whether $(S, \text{vars}(\mathcal{T}_i')) \rightarrow^\mu G$, for some $(S, \text{vars}(\mathcal{T}_i')) \in \mathbf{GtG}(\mathcal{T}_i')$.

The idea behind domination width is to ensure that $\mathbf{GtG}(\mathcal{T}_i')$ is always *dominated* by a subset $\mathcal{G} \subseteq \mathbf{GtG}(\mathcal{T}_i')$ where each generalised t-graph in $\mathcal{G}$ has small ctw. The set $\mathcal{G}$ dominates $\mathbf{GtG}(\mathcal{T}_i')$ in the sense that, for every $(S', \text{vars}(\mathcal{T}_i')) \in \mathbf{GtG}(\mathcal{T}_i')$, there is a $(S, \text{vars}(\mathcal{T}_i')) \in \mathcal{G}$ such that $(S, \text{vars}(\mathcal{T}_i')) \rightarrow (S', \text{vars}(\mathcal{T}_i'))$. Therefore, by transitivity of the relation $\rightarrow$, checking $\mu \notin [\![\mathcal{F}]\!]_G$ amounts to checking that there is a homomorphism from some element of $\mathcal{G}$ to G. Since generalised t-graphs of small ctw are well-behaved with respect to the relaxation $\rightarrow_k$ (see Proposition 3.4), this will imply that the relaxation of the natural algorithm, described at the beginning of this section, given by replacing homomorphism tests $\rightarrow$ by $\rightarrow_k$, correctly decides if $\mu \notin [\![\mathcal{F}]\!]_G$. Below we formalise this intuition.

Let $\mathcal{F} = \{\mathcal{T}_1, \ldots, \mathcal{T}_m\}$ be a wdPF. A *subtree* $\mathcal{T}$ of $\mathcal{F}$ is a subtree of some wdPT $\mathcal{T}_i$, for $i \in \{1, \ldots, m\}$. The *support* $\text{supp}(\mathcal{T})$ of the subtree $\mathcal{T}$ contains precisely the indices i from $\{1, \ldots, m\}$ such that there is a subtree $\mathcal{T}_i'$ of $\mathcal{T}_i$ satisfying $\text{vars}(\mathcal{T}_i') = \text{vars}(\mathcal{T})$. Note that $\text{supp}(\mathcal{T}) \neq \emptyset$, for every subtree $\mathcal{T}$. Since wdPTs are in NR normal form, whenever $i \in \text{supp}(\mathcal{T})$, then the witness subtree $\mathcal{T}_i'$ is unique. For $i \in \text{supp}(\mathcal{T})$, we denote such a $\mathcal{T}_i'$ by $\mathcal{T}^{\text{sp}}(i)$.

Let $\mathcal{T}$ be a subtree of $\mathcal{F} = \{\mathcal{T}_1, \ldots, \mathcal{T}_m\}$. A *children assignment* for $\mathcal{T}$ is a function Δ with a non-empty domain $\text{dom}(\Delta) \subseteq \text{supp}(\mathcal{T})$ that maps every $i \in \text{dom}(\Delta)$ to a child $\Delta(i)$ of $\mathcal{T}^{\text{sp}}(i)$. We denote by $\mathbf{CA}(\mathcal{T})$ the set of all children assignments for $\mathcal{T}$. Observe that if $\Delta \in \mathbf{CA}(\mathcal{T})$, then it must be the case that $\mathcal{T}^{\text{sp}}(i) \neq \mathcal{T}_i$, for every $i \in \text{dom}(\Delta)$. In particular, it could be the case that $\mathbf{CA}(\mathcal{T}) = \emptyset$. The *renamed t-graphs assignment* ρ_Δ associated with Δ maps $i \in \text{dom}(\Delta)$ to a t-graph $\rho_\Delta(i)$ obtained from $\text{pat}(\Delta(i))$ by renaming all variables in $\text{vars}(\Delta(i)) \setminus \text{vars}(\mathcal{T})$ to new fresh variables. In particular, if $i, j \in \text{dom}(\Delta)$ and $i \neq j$, then

$$(\text{vars}(\rho_\Delta(i)) \setminus \text{vars}(\mathcal{T})) \cap (\text{vars}(\rho_\Delta(j)) \setminus \text{vars}(\mathcal{T})) = \emptyset.$$

For $\Delta \in \mathbf{CA}(\mathcal{T})$, we define the t-graph S_Δ as

$$S_\Delta := \text{pat}(\mathcal{T}) \cup \bigcup_{i \in \text{dom}(\Delta)} \rho_\Delta(i).$$

We say that a children assignment $\Delta \in \mathbf{CA}(\mathcal{T})$ is *valid* if for every $i \in \text{supp}(\mathcal{T}) \setminus \text{dom}(\Delta)$, we have that

$$(\text{pat}(\mathcal{T}^{\text{sp}}(i)), \text{vars}(\mathcal{T})) \not\rightarrow (S_\Delta, \text{vars}(\mathcal{T})).$$

We denote by $\mathbf{VCA}(\mathcal{T})$ the set of valid children assignments for $\mathcal{T}$. Finally, for the subtree $\mathcal{T}$, we define the set of generalised t-graphs *associated with* $\mathcal{T}$ as

$$\mathbf{GtG}(\mathcal{T}) := \{(S_\Delta, \text{vars}(\mathcal{T})) \mid \Delta \in \mathbf{VCA}(\mathcal{T})\}.$$

Example 3.6. Let $k \geq 2$. Recall from Example 3.2 that

$$K_k(?o_1, \ldots, ?o_k) = \{(?o_i, r, ?o_j) \mid i, j \in \{1, \ldots, k\} \text{ with } i < j\}.$$

Consider the wdPF $\mathcal{F}_k = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$ depicted in Figure 2. For a wdPT $\mathcal{T} = (T, r, \lambda)$ and a subset $N \subseteq V(T)$, we denote by $\mathcal{T}[N]$ the subtree of $\mathcal{T}$ induced by the set of nodes N. Observe that the only subtrees $\mathcal{T}$ of $\mathcal{F}$ with a non-empty set $\mathbf{GtG}(\mathcal{T})$ are $\mathcal{T}_1[r_1]$,

[4]The existential k-pebble game is known to capture the so-called *k-consistency test* [13], which is a well-known heuristic for solving *constraint satisfaction problems* (CSPs).
[5]In [6], it was shown that $\rightarrow_k$ and $\rightarrow$ coincide for relational structures whose cores have treewidth at most $k - 1$. For Proposition 3.4, we need a generalisation of the results in [6] that considers relational structures equipped with a set of *distinguished elements*. Indeed, such distinguished elements correspond to the variables in X and the IRIs appearing in the generalised t-graph (S, X). Such a generalisation follows straightforwardly from the results in [6].

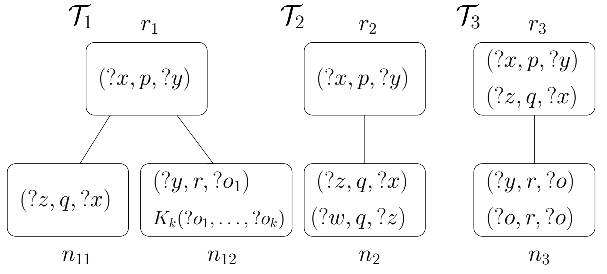

Figure 2: The wdPF $\mathcal{F}_k = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$ of Example 3.6.

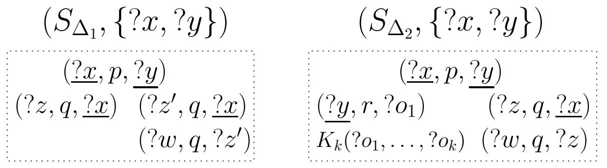
$(S_{\Delta_1}, \{?x, ?y\})$
$(\underline{?x}, p, \underline{?y})$
$(?z, q, \underline{?x})$
$(?z', q, \underline{?x})$
$(?w, q, ?z')$
$(S_{\Delta_2}, \{?x, ?y\})$
$(\underline{?x}, p, \underline{?y})$
$(\underline{?y}, r, ?o_1)$
$(?z, q, \underline{?x})$
$K_k(?o_1, \dots, ?o_k)$
$(?w, q, ?z)$

Figure 3: The generalised t-graphs $(S_{\Delta_1}, \{?x, ?y\})$ and $(S_{\Delta_2}, \{?x, ?y\})$ from Examples 3.6 and 3.9.

$\mathcal{T}_1[r_1, n_{11}]$, $\mathcal{T}_1[r_1, n_{12}]$, $\mathcal{T}_2[r_2]$ and $\mathcal{T}_3[r_3]$. Consider first $\mathcal{T}_1[r_1]$ and note that $\text{supp}(\mathcal{T}_1[r_1]) = \{1, 2\}$. We have that

$$\mathbf{GtG}(\mathcal{T}_1[r_1]) = \{(S_{\Delta_1}, \{?x, ?y\}), (S_{\Delta_2}, \{?x, ?y\})\}$$

with $\Delta_1, \Delta_2 \in \mathbf{VCA}(\mathcal{T}_1[r_1])$, where Δ_1 and Δ_2 are described by $\Delta_1 = \{1 \mapsto n_{11}, 2 \mapsto n_2\}$ and $\Delta_2 = \{1 \mapsto n_{12}, 2 \mapsto n_2\}$. Figure 3 illustrates $(S_{\Delta_1}, \{?x, ?y\})$ and $(S_{\Delta_2}, \{?x, ?y\})$. Note how we need to rename $?z$ to a fresh variable $?z'$ in $(S_{\Delta_1}, \{?x, ?y\})$. Observe also that, for instance, the children assignment given by $\Delta_3 = \{1 \mapsto n_{11}\}$ is not valid as $2 \notin \text{dom}(\Delta_3)$ and

$$(\text{pat}(\mathcal{T}_2[r_2]), \{?x, ?y\}) \to (S_{\Delta_3}, \{?x, ?y\}).$$

For $\mathcal{T}_1[r_1, n_{11}]$, we have that

$$\mathbf{GtG}(\mathcal{T}_1[r_1, n_{11}]) = \{(S_{\Delta}, \{?x, ?y, ?z\})\}$$

where $\Delta = \{1 \mapsto n_{12}, 3 \mapsto n_3\}$. Note that $(S', \{?x, ?y, ?z\})$ in Figure 1 corresponds to $(S_{\Delta}, \{?x, ?y, ?z\})$. In the case of $\mathcal{T}_1[r_1, n_{12}]$, we have that

$$\mathbf{GtG}(\mathcal{T}_1[r_1, n_{12}]) = \{(S_{\Delta'}, \{?x, ?y, ?o_1, \dots, ?o_k\})\}$$

where $\Delta' = \{1 \mapsto n_{11}\}$. Finally, note that $\mathbf{GtG}(\mathcal{T}_2[r_2]) = \mathbf{GtG}(\mathcal{T}_1[r_1])$ and $\mathbf{GtG}(\mathcal{T}_3[r_3]) = \mathbf{GtG}(\mathcal{T}_1[r_1, n_{11}])$.

Now we are ready to define domination width.

Definition 3.7 (k-domination). Let $\mathcal{G}$ be a set of generalised t-graphs of the form $\mathcal{G} = \{(S, X) \mid S \in \mathcal{S}\}$, where $\mathcal{S}$ is a set of t-graphs and X is a fixed set of variables with $X \subseteq \text{vars}(S)$, for all $S \in \mathcal{S}$. We say that $\mathcal{G}' \subseteq \mathcal{G}$ is a *dominating set* of $\mathcal{G}$ if for every $(S, X) \in \mathcal{G} \setminus \mathcal{G}'$, there exists $(S', X) \in \mathcal{G}'$ such that $(S', X) \to (S, X)$.

We say that $\mathcal{G}$ is *k-dominated* if the set $\{(S, X) \in \mathcal{G} \mid \text{ctw}(S, X) \leq k\}$ is a dominating set of $\mathcal{G}$.

Definition 3.8 (Domination width). Let $\mathcal{F}$ be a wdPF. The *domination width* of $\mathcal{F}$, denoted by $\text{dw}(\mathcal{F})$, is the minimum positive integer such that for every subtree $\mathcal{T}$ of $\mathcal{F}$, the set of generalised t-graphs $\mathbf{GtG}(\mathcal{T})$ is k-dominated.

For a well-designed graph pattern P, we define the *domination width* of P as $\text{dw}(P) := \text{dw}(\mathbf{wdpf}(P))$.

We say that a class C of well-designed graph patterns has *bounded* domination width if there is a universal constant $k \geq 1$ such that $\text{dw}(P) \leq k$, for every $P \in C$.

Example 3.9. Consider a class $C = \{P_k \mid k \geq 2\}$ such that $\mathbf{wdpf}(P_k) = \mathcal{F}_k$, where $\mathcal{F}_k$ is the wdPF defined in Figure 2 and Example 3.6. We claim that C has bounded domination width as for every $k \geq 2$, it is the case that $\text{dw}(\mathcal{F}_k) = 1$. Indeed, following the notation from Example 3.6, we need to check that $\mathbf{GtG}(\mathcal{T}_1[r_1])$, $\mathbf{GtG}(\mathcal{T}_1[r_1, n_{11}])$ and $\mathbf{GtG}(\mathcal{T}_1[r_1, n_{12}])$ are 1-dominated.

Note first that $\text{ctw}(S_{\Delta'}, \{?x, ?y, ?o_1, \dots, ?o_k\}) = 1$. Therefore, $\mathbf{GtG}(\mathcal{T}_1[r_1, n_{12}])$ is 1-dominated. Observe also that $(S_{\Delta}, \{?x, ?y, ?z\})$ coincides with $(S', \{?x, ?y, ?z\})$ from Figure 1 and, as explained in Example 3.2, we have $\text{ctw}(S', \{?x, ?y, ?z\}) = 1$. It follows that $\mathbf{GtG}(\mathcal{T}_1[r_1, n_{11}])$ is also 1-dominated. Finally, for $\mathcal{T}_1[r_1]$, we have that $\text{ctw}(S_{\Delta_1}, \{?x, ?y\}) = 1$ and $\text{ctw}(S_{\Delta_2}, \{?x, ?y\}) = k - 1$ (see Figure 3). However, we have that $(S_{\Delta_1}, \{?x, ?y\}) \to (S_{\Delta_2}, \{?x, ?y\})$, and hence, $\mathbf{GtG}(\mathcal{T}_1[r_1])$ is also 1-dominated.

The following is our main tractability result.

Theorem 3.10 (Main tractability). *Let C be a class of well-designed graph patterns of bounded domination width. Then* wdEVAL(C) *is in PTIME.*

Proof. Let $k \geq 1$ be a positive integer such that $\text{dw}(P) \leq k$, for all $P \in C$. Fix $P \in C$, RDF graph G and mapping μ. Let $\mathcal{F} := \mathbf{wdpf}(P)$ and suppose that $\mathcal{F} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$. As hinted at the beginning of this section, the idea for checking $\mu \in [\![\mathcal{F}]\!]_G$ is to apply the natural evaluation algorithm for wdPFs (see e.g. [17, 24]), but instead of checking whether μ can be extended to a child via a homomorphism, we check whether it can be extended via the existential $(k + 1)$-pebble game.

Formally, we iterate over the set $\{1, \dots, m\}$ starting from $i = 1$, and check the existence of a subtree $\mathcal{T}_i^{\mu}$ of $\mathcal{T}_i$ such that μ is a homomorphism from $\text{pat}(\mathcal{T}_i^{\mu})$ to G (in particular, $\text{vars}(\mathcal{T}_i^{\mu}) = \text{dom}(\mu)$). If there is no such a subtree, we continue with $i + 1$. By condition (3) of wdPTs, the previous check can be done in polynomial time. Note also that $\mathcal{T}_i^{\mu}$ is unique (if it exists), as $\mathcal{T}_i$ is in NR normal form. We now check that for all children n of $\mathcal{T}_i^{\mu}$, it is *not* the case that

$$(\text{pat}(\mathcal{T}_i^{\mu}) \cup \text{pat}(n), \text{vars}(\mathcal{T}_i^{\mu})) \to_{k+1}^{\mu} G.$$

If this holds, we *accept* the instance; otherwise we continue with $i + 1$. If for every $i \in \{1, \dots, m\}$ the instance is not accepted, then we *reject* the instance.

Notice that by Proposition 3.3 the above-described algorithm can be implemented in polynomial time. Observe also that the algorithm is always sound (independently of the assumption that C is of bounded domination width). Indeed, suppose that $\mu \notin [\![\mathcal{F}]\!]_G$. This means that for each $i \in \{1, \dots, m\}$, either $\mathcal{T}_i^{\mu}$ does not exist or there is such a $\mathcal{T}_i^{\mu}$ and there is a child n and a homomorphism ν from $\text{pat}(n)$ to G compatible with μ. In other words,

$$(\text{pat}(\mathcal{T}_i^{\mu}) \cup \text{pat}(n), \text{vars}(\mathcal{T}_i^{\mu})) \to^{\mu} G.$$

By property (2), we have that

$$(\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \mathrm{pat}(n), \mathrm{vars}(\mathcal{T}_i^{\mu})) \to_{k+1}^{\mu} G.$$

Hence, the algorithm rejects (as it does not accept in any iteration).

For completeness, assume that $\mu \in [\![\mathcal{F}]\!]_G$, i.e., $\mu \in [\![\mathcal{T}_\ell]\!]_G$, for some $\ell \in \{1, \dots, m\}$. In particular, there is a (unique) subtree $\mathcal{T}$ of $\mathcal{T}_\ell$ such that μ is a homomorphism from $\mathrm{pat}(\mathcal{T})$ to G and, for every child n of $\mathcal{T}$, $(\mathrm{pat}(\mathcal{T}) \cup \mathrm{pat}(n), \mathrm{vars}(\mathcal{T})) \to^{\mu} G$ does not hold. Towards a contradiction suppose that the algorithm rejects the instance $(\mathcal{F}, G, \mu)$. Let $I \subseteq \mathrm{supp}(\mathcal{T})$ be the set of indices i such that, in the i-th iteration, the algorithm finds a subtree $\mathcal{T}_i^{\mu}$ of $\mathcal{T}_i$ such that μ is a homomorphism from $\mathrm{pat}(\mathcal{T}_i^{\mu})$ to G. Observe that $I \neq \emptyset$ as $\ell \in I$. Since the algorithm rejects, we have that for every $i \in I$, there is a child n_i of the subtree $\mathcal{T}_i^{\mu}$ such that

$$(\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \mathrm{pat}(n_i), \mathrm{vars}(\mathcal{T}_i^{\mu})) \to_{k+1}^{\mu} G. \qquad (\dagger)$$

Let Δ be the children assignment with $\mathrm{dom}(\Delta) = I$ such that $\Delta(i) = n_i$, for every $i \in I$.

For readability, we let $X := \mathrm{vars}(\mathcal{T})$. We show that Δ is valid. By contradiction, suppose that there exists $j \in \mathrm{supp}(\mathcal{T}) \setminus I$ such that

$$(\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j)), X) \to (S_\Delta, X). \qquad (\ddagger)$$

Note first that $\mathrm{vars}(\mathcal{T}_i^{\mu}) = X$, for every $i \in I$, and hence

$$(\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \mathrm{pat}(n_i), \mathrm{vars}(\mathcal{T}_i^{\mu})) = (\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \mathrm{pat}(n_i), X).$$

Recall that $\rho_\Delta(i)$ is the renaming of $\mathrm{pat}(n_i)$ where variables from $\mathrm{vars}(n_i) \setminus X$ become fresh variables. We have then that, for every $i \in I$, $(\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \rho_\Delta(i), X) \to (\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \mathrm{pat}(n_i), X)$, and by ($\dagger$) and Proposition 3.5, item (1), $(\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \rho_\Delta(i), X) \to_{k+1}^{\mu} G$.

Now we can apply Proposition 3.5, item (2) to the generalised t-graphs $\{(\mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \rho_\Delta(i), X) \mid i \in I\}$ and obtain that

$$(\bigcup_{i \in I} \mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \rho_\Delta(i), X) \to_{k+1}^{\mu} G.$$

Recall that $S_\Delta = \mathrm{pat}(\mathcal{T}) \cup \bigcup_{i \in I} \rho_\Delta(i)$, and since $\ell \in I$ and $\mathcal{T} = \mathcal{T}_\ell^{\mu}$, we have $S_\Delta \subseteq \bigcup_{i \in I} \mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \rho_\Delta(i)$. It follows that

$$(S_\Delta, X) \to (\bigcup_{i \in I} \mathrm{pat}(\mathcal{T}_i^{\mu}) \cup \rho_\Delta(i), X).$$

By Proposition 3.5, item (1), we have

$$(S_\Delta, X) \to_{k+1}^{\mu} G \qquad (*)$$

and by ($\ddagger$), it follows that $(\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j)), X \to_{k+1}^{\mu} G$. As $\mathrm{vars}(\mathcal{T}^{\mathrm{sp}}(j)) \setminus X = \emptyset$, we conclude by property (1) that $(\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j)), X) \to^{\mu} G$, i.e., μ is a homomorphism from $\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j))$ to G. Since $j \notin I$, this is a contradiction with the definition of I. Thus $\Delta \in \mathbf{VCA}(\mathcal{T})$.

Since $\mathrm{dw}(\mathcal{F}) \leq k$, $\mathbf{GtG}(\mathcal{T})$ is k-dominated. In particular, it is the case that $\mathrm{ctw}(S_{\Delta'}, X) \leq k$ and $(S_{\Delta'}, X) \to (S_\Delta, X)$, for some $\Delta' \in \mathbf{VCA}(\mathcal{T})$. Proposition 3.5, item (1) and ($*$) implies $(S_{\Delta'}, X) \to_{k+1}^{\mu} G$. By Proposition 3.4, we have that $(S_{\Delta'}, X) \to^{\mu} G$. Since $\mathcal{T}^{\mathrm{sp}}(\ell) = \mathcal{T}$, we have that $(\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(\ell)), X) \to (S_{\Delta'}, X)$, and since Δ' is valid, it must be the case that $\ell \in \mathrm{dom}(\Delta')$. Observe that

$$(\mathrm{pat}(\mathcal{T}) \cup \mathrm{pat}(\Delta'(\ell)), X) \to (S_{\Delta'}, X)$$

as $S_{\Delta'}$ contains a copy of $\mathrm{pat}(\mathcal{T}) \cup \mathrm{pat}(\Delta'(\ell))$, modulo renaming of variables in $\mathrm{vars}(\Delta'(\ell)) \setminus X$. By composition, we have

$$(\mathrm{pat}(\mathcal{T}) \cup \mathrm{pat}(\Delta'(\ell)), X) \to^{\mu} G.$$

Since $\Delta'(\ell)$ is a child of $\mathcal{T}$, this contradicts the fact that $\mu \in [\![\mathcal{T}_\ell]\!]_G$. We conclude that the algorithm accepts the instance $(\mathcal{F}, G, \mu)$. □

We remark that the classes of bounded domination width *strictly* extend those that are locally tractable [17] (see also [4]), which are the most general tractable restrictions known so far. In our context, a class C is *locally tractable* if there is a constant $k \geq 1$ such that for every $P \in C$ with $\mathbf{wdpf}(P) = \mathcal{F}$, every wdPT $\mathcal{T} = (T, r, \lambda) \in \mathcal{F}$, and every node $n \in V(T)$ with $n \neq r$ and parent n', it is the case that

$$\mathrm{ctw}(\mathrm{pat}(n), \mathrm{vars}(n) \cap \mathrm{vars}(n')) \leq k.$$

Observe that local tractability implies bounded domination but the converse does not hold in general. Indeed, it suffices to consider the class $C = \{P_k \mid k \geq 2\}$ from Example 3.9. As shown in this example, C has bounded domination width but due to node n_{12} in $\mathcal{T}_1$ (see Figure 2), C is not locally tractable.

3.2 The case of UNION-free patterns

In this section, we show that for well-designed patterns using only AND and OPT, the notion of domination width boils down to a simpler notion of width called *branch treewidth.* Recall that, in this case, well-designed patterns can be represented by pattern trees, instead of pattern forests.

For a wdPT $\mathcal{T} = (T, r, \lambda)$ and $n \in V(T)$, we define the *branch* $\mathcal{B}_n$ of n to be the set of nodes in $V(T)$ appearing in the unique path in T from the root r to the parent of n. Note that $\mathcal{B}_r = \emptyset$. For $n \in V(T)$ with $n \neq r$, we define the t-graph $S_n^{\mathrm{br}} := \mathrm{pat}(n) \cup \bigcup_{n' \in \mathcal{B}_n} \mathrm{pat}(n')$ and the set of variables $X_n^{\mathrm{br}} := \mathrm{vars}(\bigcup_{n' \in \mathcal{B}_n} \mathrm{pat}(n'))$. Note that $X_n^{\mathrm{br}} \subseteq \mathrm{vars}(S_n^{\mathrm{br}})$.

Definition 3.11 (Branch treewidth). Let $\mathcal{T} = (T, r, \lambda)$ be a wdPT. We define the *branch treewidth* $\mathrm{bw}(\mathcal{T})$ of $\mathcal{T}$ to be the minimum positive integer k such that for all $n \in V(T)$ with $n \neq r$, it is the case that $\mathrm{ctw}(S_n^{\mathrm{br}}, X_n^{\mathrm{br}}) \leq k$.

For a UNION-free well-designed graph pattern P, we define the *branch treewidth* of P to be $\mathrm{bw}(P) := \mathrm{bw}(\mathcal{T})$, where $\mathcal{T}$ is the wdPT such that $\mathbf{wdpf}(P) = \{\mathcal{T}\}$.

As it turns out, branch treewidth and domination width coincide for UNION-free patterns.

Proposition 3.12. *For every UNION-free well-designed graph pattern P, we have that $dw(P) = bw(P)$.*

Proof. Assume that $\mathbf{wdpf}(P) = \{\mathcal{T}\}$, where $\mathcal{T} = (T, r, \lambda)$ is a wdPT. We start by proving that $\mathrm{dw}(\mathcal{T}) \leq \mathrm{bw}(\mathcal{T})$. Assume that $\mathrm{bw}(P) = k$ and let $\mathcal{T}'$ be a subtree of $\mathcal{T}$. We need to prove that $\mathbf{GtG}(\mathcal{T}')$ is k-dominated. We shall prove something stronger: for every $(S_\Delta, \mathrm{vars}(\mathcal{T}')) \in \mathbf{GtG}(\mathcal{T}')$, we have $\mathrm{ctw}(S_\Delta, \mathrm{vars}(\mathcal{T}')) \leq k$.

Let $(S_\Delta, \mathrm{vars}(\mathcal{T}')) \in \mathbf{GtG}(\mathcal{T}')$, where $\Delta \in \mathbf{VCA}(\mathcal{T}')$. Observe that S_Δ coincides with $S' := \mathrm{pat}(\mathcal{T}') \cup \mathrm{pat}(n)$ modulo renaming of variables in $\mathrm{vars}(n) \setminus \mathrm{vars}(\mathcal{T}')$, where n is a child of $\mathcal{T}'$. Thus $\mathrm{ctw}(S_\Delta, \mathrm{vars}(\mathcal{T}')) = \mathrm{ctw}(S', \mathrm{vars}(\mathcal{T}'))$. Note that $n \neq r$. Let (C, X_n^{br}) be the core of $(S_n^{\mathrm{br}}, X_n^{\mathrm{br}})$. In particular, (C, X_n^{br}) is a subgraph of $(S_n^{\mathrm{br}}, X_n^{\mathrm{br}})$ and $(S_n^{\mathrm{br}}, X_n^{\mathrm{br}}) \to (C, X_n^{\mathrm{br}})$. As $\mathcal{B}_n \subseteq V(T')$, where $\mathcal{T}' = (T', r, \lambda')$, we have that $(\mathrm{pat}(\mathcal{T}') \cup C, \mathrm{vars}(\mathcal{T}'))$ is a subgraph of $(S', \mathrm{vars}(\mathcal{T}'))$ and

$$(S', \mathrm{vars}(\mathcal{T}')) \to (\mathrm{pat}(\mathcal{T}') \cup C, \mathrm{vars}(\mathcal{T}')).$$

Then the core of $(S', \mathrm{vars}(\mathcal{T}'))$ is a subgraph of $(\mathrm{pat}(\mathcal{T}')\cup C, \mathrm{vars}(\mathcal{T}'))$. As treewidth does not increase by taking subgraphs, and using the fact that

$$\mathrm{tw}(\mathrm{pat}(\mathcal{T}') \cup C, \mathrm{vars}(\mathcal{T}')) = \mathrm{tw}(C, X_n^{\mathrm{br}})$$

as their Gaifman graphs coincide, we have that

$$\mathrm{ctw}(S_\Delta, \mathrm{vars}(\mathcal{T}')) = \mathrm{ctw}(S', \mathrm{vars}(\mathcal{T}')) \leq \mathrm{tw}(C, X_n^{\mathrm{br}}).$$

Since $\mathrm{bw}(\mathcal{T}) = k$ and $n \neq r$, we have that $\mathrm{tw}(C, X_n^{\mathrm{br}}) \leq k$, and hence, $\mathrm{ctw}(S_\Delta, \mathrm{vars}(\mathcal{T}')) \leq k$ as required.

We now prove that $\mathrm{bw}(\mathcal{T}) \leq \mathrm{dw}(\mathcal{T})$. Let $\mathrm{dw}(\mathcal{T}) = k$. By contradiction, suppose that there exists $n \in V(T)$ with $n \neq r$ such that $\mathrm{ctw}(S_n^{\mathrm{br}}, X_n^{\mathrm{br}}) > k$. Let $\mathcal{T}'$ be the subtree of $\mathcal{T}$ corresponding to $\mathcal{B}_n$. In particular, n is a child of $\mathcal{T}'$ and

$$(\mathrm{pat}(\mathcal{T}') \cup \mathrm{pat}(n), \mathrm{vars}(\mathcal{T}')) = (S_n^{\mathrm{br}}, X_n^{\mathrm{br}}).$$

For readability, we let $S := S_n^{\mathrm{br}}$ and $X' := X_n^{\mathrm{br}} = \mathrm{vars}(\mathcal{T}')$. Since $\mathbf{GtG}(\mathcal{T}')$ is k-dominated and $\mathrm{ctw}(S, X') > k$, there exists a child n' of $\mathcal{T}'$ with $n' \neq n$ such that

$$(S_{n'}, X') \to (S, X') \quad (\dagger)$$

where $S_{n'} := \mathrm{pat}(\mathcal{T}') \cup \mathrm{pat}(n')$ and $\mathrm{ctw}(S_{n'}, X') \leq k$. Let $\mathcal{T}''$ be the subtree of $\mathcal{T}$ obtained from $\mathcal{T}'$ by adding the child n'. Let $S_{nn'} := \mathrm{pat}(\mathcal{T}'') \cup \mathrm{pat}(n)$ and $X'' := \mathrm{vars}(\mathcal{T}'')$. Below we show that

$$\mathrm{ctw}(S_{nn'}, X'') > k. \quad (*)$$

Towards a contradiction, assume that $\mathrm{ctw}(S_{nn'}, X'') \leq k$. We shall show that $\mathrm{ctw}(S, X') \leq k$, which is a contradiction. It is a known fact (see [6, Theorem 12]) that $\mathrm{ctw}(S, X') \leq k$ if and only if (‡) there exists (S^*, X') such that

- $\mathrm{tw}(S^*, X') \leq k$, and
- $(S, X') \to (S^*, X')$ and $(S^*, X') \to (S, X')$. In this case, we write $(S, X') \leftrightarrows (S^*, X')$.

Also, observe that (†) implies that $(S_{nn'}, X') \to (S, X')$. As $S \subseteq S_{nn'}$, we have $(S, X') \to (S_{nn'}, X')$, and hence $(S_{nn'}, X') \leftrightarrows (S, X')$. By transitivity of $\to$, it suffices to show (‡) with respect to $S_{nn'}$ instead of S.

By hypothesis, we have $\mathrm{ctw}(S_{n'}, X') \leq k$ and $\mathrm{ctw}(S_{nn'}, X'') \leq k$. Hence $\mathrm{tw}(C_{n'}, X') \leq k$ and $\mathrm{tw}(C_{nn'}, X'') \leq k$, where $(C_{n'}, X')$ and $(C_{nn'}, X'')$ are the cores of $(S_{n'}, X')$ and $(S_{nn'}, X'')$, respectively. We define the following generalised t-graphs:

$$\begin{aligned} D_{n'} &:= C_{n'} \setminus \mathrm{pat}(\mathcal{T}'), \\ D_{nn'} &:= C_{nn'} \setminus \mathrm{pat}(\mathcal{T}''), \\ S^* &:= \mathrm{pat}(\mathcal{T}') \cup D_{n'} \cup D_{nn'}. \end{aligned}$$

Note that $\mathrm{vars}(D_{n'}) \cap \mathrm{vars}(D_{nn'}) \subseteq X'$. In particular, the Gaifman graph of (S^*, X') is the disjoint union of those of $(C_{n'}, X')$ and $(C_{nn'}, X'')$, and then $\mathrm{tw}(S^*, X') \leq k$. It suffices to show that $(S_{nn'}, X') \leftrightarrows (S^*, X')$.

Observe first that $(S^*, X') \to (S_{nn'}, X')$ as $S^* \subseteq S_{nn'}$. For the other direction, observe that $(S_{nn'}, X'') \to (C_{nn'}, X'')$ by definition of cores. In particular, $(S_{nn'}, X') \to (C_{nn'}, X')$. Also by definition of cores, we have that $(S_{n'}, X') \to (C_{n'}, X')$ via a homomorphism h. Hence, by construction of S^*, the function $g : \mathrm{vars}(C_{nn'}) \to \mathbf{I} \cup \mathbf{V}$ such that $g(?x) = h(?x)$, for $?x \in \mathrm{vars}(n') \setminus X'$, and $g(?x) = ?x$ otherwise, is a homomorphism witnessing $(C_{nn'}, X') \to (S^*, X')$. By transitivity, $(S_{nn'}, X') \to (S^*, X')$ as required. Thus claim (*) holds.

As $\mathbf{GtG}(\hat{\mathcal{T}})$ is k-dominated for every subtree $\hat{\mathcal{T}}$ of $\mathcal{T}$, we can iterate the previous argument until we find a subtree $\mathcal{T}^*$ of $\mathcal{T}$ such that n is its only child and $\mathrm{ctw}(\mathrm{pat}(\mathcal{T}^*) \cup \mathrm{pat}(n), \mathrm{vars}(\mathcal{T}^*)) > k$. It follows that $\mathbf{GtG}(\mathcal{T}^*)$ cannot be k-dominated; a contradiction. □

Proposition 3.12 tells us that if $\mathcal{T}$ is a wdPT with $\mathrm{dw}(\mathcal{T}) \leq k$, then for every subtree $\mathcal{T}'$ of $\mathcal{T}$, the set $\mathbf{GtG}(\mathcal{T}')$ is k-dominated due to the trivial reason: all elements of $\mathbf{GtG}(\mathcal{T}')$ are already of ctw $\leq k$. Observe that this is not the case for arbitrary patterns. Indeed, as Example 3.9 shows, $\mathrm{dw}(\mathcal{F}_k) = 1$ but the set $\mathbf{GtG}(\mathcal{T}_1[r_1])$ is not trivially 1-dominated as $\mathrm{ctw}(S_{\Delta_2}, \{?x, ?y\}) = k - 1$.

The results in this paper (see Theorem 4.2 and Corollary 4.3 in the next section) show that domination width (and then branch treewidth for UNION-free patterns) captures tractability for well-designed patterns. Therefore, polynomial-time solvability of arbitrary patterns and UNION-free patterns is based on two different principles: for arbitrary patterns is based on k-domination, while for UNION-free patterns *branch tractability* suffices. As a matter of fact, this striking difference between the general and UNION-free case is also present in other contexts: for instance, *containment* of UNION-free patterns can be characterised in very simple terms, while the general case requires more involved characterisations (see e.g. [24, Theorem 3.7] and [14, Lemma 1]).

Finally, observe that bounded branch treewidth implies local tractability, but the converse is not true in general. Hence, we obtain new tractable classes even in the UNION-free case. To see this, consider for instance the class $C = \{P_k' \mid k \geq 2\}$, where $\mathbf{wdpf}(P_k') = \{\mathcal{T}_k'\}$, where $\mathcal{T}_k' = (T, r, \lambda)$ is a wdPT such that

- $T = (\{r, n_k\}, \{\{r, n_k\}\})$, i.e., T is the tree containing two nodes.
- $\lambda(r) = \{(?y, r, ?y)\}$ and

$$\lambda(n_k) = \{(?y, r, ?o_1)\} \cup K_k(?o_1, \ldots, ?o_k)$$

where $K_k(?o_1, \ldots, ?o_k)$ is defined as in Example 3.2.

We have that C has bounded branch treewidth as $\mathrm{bw}(\mathcal{T}_k') = 1$, for every $k \geq 2$. Indeed, the core of $(S_{n_k}^{\mathrm{br}}, X_{n_k}^{\mathrm{br}})$ is simply $(\{(?y, r, ?y)\}, \{?y\})$. On the other hand, C is not locally tractable as $\mathrm{ctw}(\mathrm{pat}(n_k), \{?y\}) = k - 1$.

4 A MATCHING HARDNESS RESULT

We start by giving some basic definitions from parameterised complexity theory as our hardness result relies on it (we refer the reader to [8] for more details).

A *parameterised problem* (Π, κ) is a classical decision problem Π equipped with a *parameterisation* κ that maps instances of Π to natural numbers. The class *FPT* contains all parameterised problems (Π, κ) that are *fixed-parameter tractable*, that is, that can be solved in time $f(\kappa(x)) \cdot |x|^{O(1)}$, where $|x|$ denotes the size of the instance and $f : \mathbb{N} \to \mathbb{N}$ is a computable function. An *fpt-reduction* from (Π, κ) to (Π', κ') is a function r mapping instances of Π to instances of Π' such that (i) for all instance x of Π, we have $x \in \Pi$ if and only if $r(x) \in \Pi'$, (ii) r can be computed in time $f(\kappa(x)) \cdot |x|^{O(1)}$ for some computable function $f : \mathbb{N} \to \mathbb{N}$, and (iii) there is a computable function $g : \mathbb{N} \mapsto \mathbb{N}$ such that for all instances x of Π, we have $\kappa'(r(x)) \leq g(\kappa(x))$.

The class W[1] can be seen as an analogue of NP in parameterised complexity theory (for a precise definition, see [8]). Proving W[1]-hardness (under fpt-reductions) is a strong indication that the problem is not in FPT as it is believed that FPT $\neq$ W[1]. A canonical W[1]-complete problem is p-CLIQUE, that is, the CLIQUE problem parameterised by the size of the clique. Recall that the CLIQUE problem asks, given an undirected graph H and a positive integer k, whether H contains a clique of size k.

Given a class C of well-designed graph patterns, we denote by p-wdEVAL(C), the problem wdEVAL(C) parameterised by the size $|P|$ of the input well-designed graph pattern P. We denote by co-wdEVAL(C) the complement of wdEVAL(C), i.e., the problem of checking $\mu \notin [\![P]\!]_G$ for a given well-designed pattern P, an RDF graph G and a mapping μ. Similarly, we denote by p-co-wdEVAL(C) the complement of p-wdEVAL(C).

4.1 Hardness result and main characterisation theorem

Our main hardness result is as follows.

Theorem 4.1 (Main hardness). *Let C be a recursively enumerable class of well-designed graph patterns of unbounded domination width. Then p-co-wdEVAL(C) is W[1]-hard.*

We provide a proof of Theorem 4.1 in the next section. We now explain how Theorem 3.10 and 4.1 imply the main characterisation result of this paper.

Theorem 4.2 (Main). *Assume FPT $\neq$ W[1]. Let C be a recursively enumerable[6] class of well-designed graph patterns. Then, the following are equivalent:*

1. *wdEVAL(C) is in PTIME.*
2. *p-wdEVAL(C) is in FPT.*
3. *C has bounded domination width.*

Proof. (1)$\Rightarrow$(2) is immediate. For (2)$\Rightarrow$(3), if p-wdEVAL(C) is in FPT, then p-co-wdEVAL(C) also is. Then, by our assumption FPT $\neq$ W[1], p-co-wdEVAL(C) cannot be W[1]-hard. Therefore, C has bounded domination width, otherwise we reach a contradiction by Theorem 4.1. The implication (3)$\Rightarrow$(1) follows directly from Theorem 3.10. □

As a corollary of Proposition 3.12, we have the following.

Corollary 4.3. *Assume FPT $\neq$ W[1]. Let C be a recursively enumerable class of UNION-free well-designed graph patterns. Then, the following are equivalent:*

1. *wdEVAL(C) is in PTIME.*
2. *p-wdEVAL(C) is in FPT.*
3. *C has bounded branch treewidth.*

[6]As in [9], we can remove the assumption of C being recursively enumerable by assuming a stronger assumption than FPT $\neq$ W[1] involving non-uniform complexity classes.

4.2 Proof of Theorem 4.1

We follow a similar strategy of the classical result by Grohe [9] that shows W[1]-hardness for evaluating a class C of CQs over schemas of *bounded arity* whose cores have unbounded treewidth. As in [9], we exhibit an fpt-reduction from p-CLIQUE to p-co-wdEVAL(C) exploiting the Excluded Grid Theorem [28] that states that there exists a function $\mathtt{w} : \mathbb{N} \to \mathbb{N}$ such that for every $k \geq 1$, the $(k \times k)$-*grid* is a *minor* of every graph of treewidth at least $\mathtt{w}(k)$ (see [7] for technical details). Throughout this section, we use $\mathtt{w}$ to denote such a function.

The first ingredient in our proof is the following variant of the main construction from [9] to take distinguished elements into account. (See the appendix for a proof.)

Lemma 4.4. *Let $k \geq 2$ and H be an undirected graph. Let (S, X) be a generalised t-graph with $ctw(S, X) \geq \mathtt{w}(\binom{k}{2})$. Then there is a generalised t-graph (B, X) such that*

1. *if $t \in S$ and $vars(t) \subseteq X$, then $t \in B$.*
2. *$(B, X) \to (S, X)$.*
3. *H contains a clique of size k iff $(S, X) \to (B, X)$.*
4. *(B, X) can be computed in time $f(k, |(S, X)|) \cdot |H|^{O(1)}$, where $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is a computable function.*

The second ingredient is the following basic property of wdPFs of large domination width. Intuitively, it states that every wdPF $\mathcal{F}$ of large domination width contains a subtree $\mathcal{T}$ with an associated generalised t-graph $(S, X) \in \mathbf{GtG}(\mathcal{T})$ of large ctw(S, X), satisfying a particular minimality condition.

Lemma 4.5. *Let $k \geq 2$ and $\mathcal{F}$ be a wdPF such that $dw(\mathcal{F}) \geq k$. Then there exists a subtree $\mathcal{T}$ of $\mathcal{F}$, and $(S, vars(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$ such that*

1. *$ctw(S, vars(\mathcal{T})) \geq k$, and*
2. *$(S', vars(\mathcal{T})) \to (S, vars(\mathcal{T})) \Rightarrow (S, vars(\mathcal{T})) \to (S', vars(\mathcal{T}))$, for every $(S', vars(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$.*

Proof. Suppose that dw$(\mathcal{F}) \geq k$, i.e., dw$(\mathcal{F}) \leq k - 1$ does not hold. By definition of domination width, there is a subtree $\mathcal{T}$ of $\mathcal{F}$ such that $\mathbf{GtG}(\mathcal{T})$ is not $(k-1)$-dominated. In particular, the following subset $\mathcal{G} \subseteq \mathbf{GtG}(\mathcal{T})$ is non-empty: $(R, \text{vars}(\mathcal{T})) \in \mathcal{G}$ if and only if $(R, \text{vars}(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$, ctw$(R, \text{vars}(\mathcal{T})) \geq k$, and $(R', \text{vars}(\mathcal{T})) \nrightarrow (R, \text{vars}(\mathcal{T}))$, for all $(R', \text{vars}(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$ with ctw$(R', \text{vars}(\mathcal{T})) \leq k-1$. Consider the directed graph H with vertex set $\mathcal{G}$ and the existence of homomorphism relation $\to$ as the edge relation. Let C be a minimal strongly connected component of H and pick any $(S, \text{vars}(\mathcal{T})) \in C$. We claim that $(S, \text{vars}(\mathcal{T}))$ satisfies the required conditions. Indeed, suppose that $(S', \text{vars}(\mathcal{T})) \to (S, \text{vars}(\mathcal{T}))$. Since $(S, \text{vars}(\mathcal{T})) \in \mathcal{G}$, and by construction of $\mathcal{G}$, it must be the case that $(S', \text{vars}(\mathcal{T})) \in \mathcal{G}$. Since C is minimal, $(S', \text{vars}(\mathcal{T})) \in C$, and then there is a directed path from $(S, \text{vars}(\mathcal{T}))$ to $(S', \text{vars}(\mathcal{T}))$ in H. By transitivity of the relation $\to$, we have $(S, \text{vars}(\mathcal{T})) \to (S', \text{vars}(\mathcal{T}))$ as required. □

The reduction. We now present an fpt-reduction from p-CLIQUE to p-co-wdEVAL(C). Let $k \geq 2$ and H be an undirected graph. We start by enumerating the class C until we find some $P \in C$ such that dw$(P) \geq \mathtt{w}(\binom{k}{2})$. Since C has unbounded domination width, this is always possible. Since the domination width is computable, we can find P in time $\alpha(k)$, for a computable function $\alpha : \mathbb{N} \to \mathbb{N}$. Let $\mathcal{F} := \mathbf{wdpf}(P)$. Since $\mathrm{dw}(\mathcal{F}) \geq \mathsf{w}(\binom{k}{2})$, we can apply Lemma 4.5 to obtain a subtree $\mathcal{T}$ of $\mathcal{F}$ and $(S, \mathrm{vars}(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$ satisfying the conditions of the lemma. By condition (1), $\mathrm{ctw}(S, \mathrm{vars}(\mathcal{T})) \geq \mathsf{w}(\binom{k}{2})$ and hence, by Lemma 4.4, we can compute in time $f(k, |(S, \mathrm{vars}(\mathcal{T}))|) \cdot |H|^{O(1)}$ a generalised t-graph $(B, \mathrm{vars}(\mathcal{T}))$ satisfying the conditions in the lemma. Observe that $(S, \mathrm{vars}(\mathcal{T}))$ only depends on k and thus, $(B, \mathrm{vars}(\mathcal{T}))$ can be computed in time $g(k) \cdot |H|^{O(1)}$, for some computable function g.

Now we define an RDF graph G and a mapping μ with $\mathrm{dom}(\mu) = \mathrm{vars}(\mathcal{T})$. The idea is that G is precisely B but interpreted as an RDF graph, i.e., we freeze the variables of B, which now become IRIs, and μ is the identity mapping over $\mathrm{vars}(\mathcal{T})$, modulo freezing of variables in B (note that $\mathrm{vars}(\mathcal{T}) \subseteq \mathrm{vars}(B)$). Formally, for $?x \in \mathrm{vars}(B)$, we define $a_{?x}$ to be an IRI. We define $\Psi : \mathrm{vars}(B) \to \mathbf{I}$ to be the mapping that maps each $?x \in \mathrm{vars}(B)$ to $a_{?x}$. Let G be the RDF graph defined by the set $G := \{\Psi(t) \mid t \in B\}$ and let μ be the mapping with $\mathrm{dom}(\mu) = \mathrm{vars}(\mathcal{T})$ such that $\mu(?x) = \Psi(?x)$, for every $?x \in \mathrm{vars}(\mathcal{T})$. By construction, Ψ is a homomorphism from B to G and $(B, \mathrm{vars}(\mathcal{T})) \to^{\mu} G$. We also define a function $\Theta : \mathrm{dom}(G) \to \mathbf{I} \cup \mathbf{V}$, where $\mathrm{dom}(G) \subseteq \mathbf{I}$ is the set of IRIs appearing in G, such that $\Theta(a) = ?x$ if $a = a_{?x}$ and $\Theta(a) = a$ otherwise.

Observe that $|P| \leq \alpha(k)$ and that (P, G, μ) can be computed in fpt-time from (H, k), that is, in time $g'(k) \cdot |H|^{O(1)}$ for some computable function g'. It remains to show that our reduction is correct, that is, H contains a clique of size k if and only if $\mu \notin [\![P]\!]_G = [\![\mathcal{F}]\!]_G$.

Correctness of the reduction. Suppose first that H contains a clique of size k. Assume $\mathcal{F} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ and $(S, \mathrm{vars}(\mathcal{T})) = (S_\Delta, \mathrm{vars}(\mathcal{T}))$, for some $\Delta \in \mathbf{VCA}(\mathcal{T})$. Let $\mathcal{T}'_\ell$ be a subtree of $\mathcal{T}_\ell$, with $\ell \in \{1, \dots, m\}$, such that μ is a homomorphism from $\mathrm{pat}(\mathcal{T}'_\ell)$ to G. We claim that there is a child n of $\mathcal{T}'_\ell$ such that

$$(\mathrm{pat}(\mathcal{T}'_\ell) \cup \mathrm{pat}(n), \mathrm{vars}(\mathcal{T}'_\ell)) \to^{\mu} G.$$

Note that this implies that $\mu \notin [\![\mathcal{T}_\ell]\!]_G$, and since ℓ is arbitrary, it follows that $\mu \notin [\![\mathcal{F}]\!]_G$ as required. We prove first that $\ell \in \mathrm{dom}(\Delta)$. Note that $\Theta \circ \mu$ is a homomorphism from $\mathrm{pat}(\mathcal{T}'_\ell)$ to B. By definition of μ, we have that $(\mathrm{pat}(\mathcal{T}'_\ell), \mathrm{vars}(\mathcal{T})) \to (B, \mathrm{vars}(\mathcal{T}))$. By item (2) in Lemma 4.4, it follows that $(\mathrm{pat}(\mathcal{T}'_\ell), \mathrm{vars}(\mathcal{T})) \to (S_\Delta, \mathrm{vars}(\mathcal{T}))$. Since $\mathcal{T}'_\ell = \mathcal{T}^{\mathrm{sp}}(\ell)$ and Δ is valid, it must be the case that $\ell \in \mathrm{dom}(\Delta)$.

Recall that $S_\Delta = \mathrm{pat}(\mathcal{T}) \cup \bigcup_{i \in \mathrm{dom}(\Delta)} \rho_\Delta(i)$, where $\rho_\Delta(i)$ is obtained from $\mathrm{pat}(\Delta(i))$ by renaming the variables in $\mathrm{vars}(\Delta(i)) \setminus \mathrm{vars}(\mathcal{T})$ to fresh variables. Since H contains a clique of size k, we obtain from Lemma 4.4, item (3) that $(S_\Delta, \mathrm{vars}(\mathcal{T})) \to (B, \mathrm{vars}(\mathcal{T}))$. Since $(B, \mathrm{vars}(\mathcal{T})) \to^{\mu} G$, we have that $(S_\Delta, \mathrm{vars}(\mathcal{T})) \to^{\mu} G$. In particular, there is a homomorphism ν from $\rho_\Delta(\ell)$ to G compatible with μ. It follows that there is a homomorphism ν' from $\mathrm{pat}(\Delta(\ell))$ to G compatible with μ. By considering $\mu \cup \nu'$, we have that

$$(\mathrm{pat}(\mathcal{T}'_\ell) \cup \mathrm{pat}(\Delta(\ell)), \mathrm{vars}(\mathcal{T})) \to^{\mu} G.$$

As $\Delta(\ell)$ is a child of $\mathcal{T}^{\mathrm{sp}}(\ell) = \mathcal{T}'_\ell$, the claim follows. Thus $\mu \notin [\![\mathcal{F}]\!]_G$.

Assume now that $\mu \notin [\![\mathcal{F}]\!]_G$. Let $I \subseteq \mathrm{supp}(\mathcal{T})$ such that $i \in I$ if and only if μ is a homomorphism from $\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(i))$ to G. We claim that $I \neq \emptyset$. Since $\mathcal{T}$ is a subtree of $\mathcal{F}$, it suffices to show that μ is a homomorphism from $\mathrm{pat}(\mathcal{T})$ to G. To see this, let $t \in \mathrm{pat}(\mathcal{T})$. In particular, $t \in S_\Delta$ and $\mathrm{vars}(t) \subseteq \mathrm{vars}(\mathcal{T})$. We can invoke item (1) in Lemma 4.4 and obtain that $t \in B$. By definition of G, $\Psi(t) \in G$, and since $\mu(t) = \Psi(t)$, it follows that $\mu(t) \in G$. Then μ is a homomorphism from $\mathrm{pat}(\mathcal{T})$ to G and $I \neq \emptyset$.

Since $\mu \notin [\![\mathcal{F}]\!]_G$, for every $i \in I$, there exists a child n_i of $\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(i))$ and a homomorphism ν_i from $\mathrm{pat}(n_i)$ to G compatible with μ. Let Δ' be the children assignment with $\mathrm{dom}(\Delta') = I$ such that $\Delta'(i) = n_i$, for every $i \in I$. It follows that, for every $i \in I$, there is a homomorphism ν'_i from $\rho_{\Delta'}(i)$ to G compatible with μ. By definition of $S_{\Delta'}$, the mapping $h = \mu \cup \bigcup_{i \in I} \nu'_i$ is well-defined and is a homomorphism from $S_{\Delta'}$ to G. In particular, $(S_{\Delta'}, \mathrm{vars}(\mathcal{T})) \to^{\mu} G$. We now show that Δ' is valid. By contradiction, assume that there is $j \in \mathrm{supp}(\mathcal{T}) \setminus I$ such that $(\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j)), \mathrm{vars}(\mathcal{T})) \to (S_{\Delta'}, \mathrm{vars}(\mathcal{T}))$. Since $(S_{\Delta'}, \mathrm{vars}(\mathcal{T})) \to^{\mu} G$, we have that

$$(\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j)), \mathrm{vars}(\mathcal{T})) \to^{\mu} G.$$

In particular, μ is a homomorphism from $\mathrm{pat}(\mathcal{T}^{\mathrm{sp}}(j))$ to G, which contradicts the definition of I. Hence $\Delta' \in \mathbf{VCA}(\mathcal{T})$ and consequently $(S_{\Delta'}, \mathrm{vars}(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$.

Observe that, by considering $\Theta \circ h$, $(S_{\Delta'}, \mathrm{vars}(\mathcal{T})) \to (B, \mathrm{vars}(\mathcal{T}))$. By item (2) of Lemma 4.4, $(B, \mathrm{vars}(\mathcal{T})) \to (S_\Delta, \mathrm{vars}(\mathcal{T}))$, and hence, $(S_{\Delta'}, \mathrm{vars}(\mathcal{T})) \to (S_\Delta, \mathrm{vars}(\mathcal{T}))$. Since $(S_{\Delta'}, \mathrm{vars}(\mathcal{T})) \in \mathbf{GtG}(\mathcal{T})$ and by item (2), Lemma 4.5, we have that $(S_\Delta, \mathrm{vars}(\mathcal{T})) \to (S_{\Delta'}, \mathrm{vars}(\mathcal{T}))$, and then $(S_\Delta, \mathrm{vars}(\mathcal{T})) \to (B, \mathrm{vars}(\mathcal{T}))$. We can apply item (3) of Lemma 4.4 and conclude that H contains a clique of size k as required.

5 CONCLUSIONS

We have introduced the notion of domination width for well-designed graph patterns. We showed that patterns with bounded domination width can be evaluated in polynomial time (Theorem 3.10). In a matching hardness result, we showed that classes of unbounded domination width cannot be evaluated in polynomial time (Theorem 4.1), unless a widely believed assumption from parameterised complexity fails. This provides a complete complexity classification for the evaluation problem restricted to admissible classes of well-designed graph patterns (Theorem 4.2).

A possible direction for future work is to additionally consider the FILTER and SELECT operators (for a formal semantics of these operators, we refer the reader to [23, 24]). We remark, however, that a complete characterisation of the tractable restrictions seems challenging in these cases. Indeed, observe that our classification of Theorem 4.2 is based on the following dichotomy: either co-wdEVAL(C) is in PTIME or it is W[1]-hard. As we explain below, it is known that this dichotomy fails if we add FILTER or SELECT, in the sense that there is a class C of queries such that co-wdEVAL(C) is in FPT but is NP-hard.

For the case of FILTER, we note that well-designed patterns using the FILTER operator can express CQs with *inequalities*. Consequently, for each class of undirected graphs $\mathcal{H}$, it is possible to construct a class $C_{\mathcal{H}}$ of well-designed patterns using AND, OPT and FILTER such that co-wdEVAL($C_{\mathcal{H}}$) is polynomial-time equivalent to the *embedding* problem EMB($\mathcal{H}$) for $\mathcal{H}$. In EMB($\mathcal{H}$), we are given two undirected graphs H and H', where $H \in \mathcal{H}$, and the question is whether there is an embedding, i.e., an injective homomorphism from H to H'. It is known, for instance, that EMB($\mathcal{P}$) (and consequently co-wdEVAL($C_{\mathcal{P}}$)) is in FPT but is NP-hard, where $\mathcal{P}$

is the class of all paths (see e.g. [9, Section 8] and [8, Section 13.3] for more details).

For SELECT (or *projection*), it was recently shown in [16] that the evaluation problem for the so-called classes of patterns using AND, OPT and SELECT of bounded *global* treewidth and *semi-bounded interface* is in FPT (see [16, Theorem 5]) but NP-hard (as pointed out in [16], NP-hardness already follows from results in [4]).

While the above discussion suggests that obtaining a precise characterisation of the tractable classes in the presence of FILTER or SELECT could be difficult, an interesting research direction would be to characterise the classes that are fixed-parameter tractable. In a recent unpublished manuscript [21], this problem was studied for (not necessarily well-designed) pattern trees with projection and several complexity classifications were obtained. Their work differs to ours in that they consider more expressive patterns and aim for fixed-parameter tractability while we consider simpler patterns but deal with polynomial-time tractability. Regarding the FILTER operator, let us remark that obtaining characterisations for fixed-parameter tractability in the presence of FILTER would require to solve a known open problem, namely, the corresponding characterisation for problems of the form EMB($\mathcal{H}$) (for further details and recent results, see e.g. [9, 18, 29]).

It would be also interesting to obtain similar structural characterisations for other variants of the evaluation problem such as the problem of *counting* the number of solutions or *enumerating* all solutions (see e.g. [16, 27]); or for fragments beyond the well-designed one such as the class of *weakly* well-designed queries [11].

Finally, note that, related to our results, we have the *recognisition* problem: given a well-designed graph pattern P, decide whether $\text{dw}(P) \leq k$ (we assume $k \geq 1$ to be fixed). Observe that Proposition 3.12 gives us an NP upper bound for this problem in the case of UNION-free patterns (as checking bw $\leq k$ is in NP). Also, by using the fact that checking whether a relational structure has a core of treewidth at most k is NP-complete [6, Theorem 13], we obtain that the recognition problem for UNION-free patterns is actually NP-complete. For arbitrary well-designed graph patterns, it is possible to obtain a Π_2^p upper bound from the definition of domination width. It remains an open question whether this Π_2^p bound is tight.

REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases.* Addison-Wesley, 1995.
[2] R. Angles and C. Gutiérrez. The expressive power of SPARQL. In *ISWC*, pages 114–129, 2008.
[3] M. Arenas, S. Conca, and J. Perez. Counting beyond a Yottabyte, or how SPARQL 1.1 property paths will prevent adoption of the standard. In *WWW*, pages 629–638, 2012.
[4] P. Barceló, R. Pichler, and S. Skritek. Efficient evaluation and approximation of well-designed pattern trees. In *PODS*, pages 131–144, 2015.
[5] M. Chandra and P. Merlin. Optimal implementation of conjunctive queries in relational databases. In *STOC*, pages 77–90, 1977.
[6] V. Dalmau, P. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *CP*, pages 310–326, 2002.
[7] R. Diestel. *Graph theory.* Springer, 2010.
[8] J. Flum and M. Grohe. *Parameterized complexity theory.* Springer, 2006.
[9] M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *Journal of the ACM*, 54(1):38–74, 2007.
[10] P. Hell and J. Nešetřil. The core of a graph. *Discrete Mathematics*, 109(1):117 – 126, 1992.
[11] M. Kaminski and E. Kostylev. Beyond well-designed SPARQL. In *ICDT*, pages 5:1–5:18, 2016.
[12] P. Kolaitis and M. Y. Vardi. On the expressive power of Datalog: Tools and a case study. *Journal of Computer and System Sciences*, 51:110–134, 1995.
[13] P. Kolaitis and M. Y. Vardi. A game-theoretic approach to constraint satisfaction. In *AAAI*, pages 175–181, 2000.
[14] E. Kostylev, J. Reutter, M. Romero, and D. Vrgoc. SPARQL with property paths. In *ISWC*, pages 3–18, 2015.
[15] E. Kostylev, J. L. Reutter, and M. Ugarte. CONSTRUCT queries in SPARQL. In *ICDT*, pages 212–229, 2015.
[16] M. Kroll, R. Pichler, and S. Skritek. On the complexity of enumerating the answers to well-designed pattern trees. In *ICDT*, pages 22:1–22:18, 2016.
[17] A. Letelier, J. Pérez, R. Pichler, and S. Skritek. Static analysis and optimization of Semantic Web queries. *ACM Trans. on Database Systems*, 38(4), 2013.
[18] B. Lin. The parameterized complexity of k-biclique. In *SODA*, pages 605–615, 2015.
[19] K. Losemann and W. Martens. The complexity of regular expressions and property paths in SPARQL. *ACM Trans. Database Syst.*, 38(4):24:1–24:39, 2013.
[20] F. Manola and E. Miller. RDF Primer. W3C Recommendation, 10 February 2004. http://www.w3.org/tr/2004/rec-rdf-primer-20040210/.
[21] S. Mengel and S. Skritek. On tractable query evaluation for SPARQL. December 2017. arXiv:1712.08939.
[22] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. In *ISWC*, pages 30–43, 2006.
[23] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. *ACM Trans. on Database Systems*, 34(3), 2009.
[24] R. Pichler and S. Skritek. Containment and equivalence of well-designed SPARQL. In *PODS*, pages 39–50, 2014.
[25] A. Polleres and J. P. Wallner. On the relation between SPARQL 1.1 and answer set programming. *Journal of Applied Non-Classical Logics*, 23(1–2):159–212, 2013.
[26] E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF, W3C recommendation, January 2008, `http://www.w3.org/tr/rdf-sparql-query`.
[27] S. S. R. Pichler. On the hardness of counting the solutions of SPARQL queries. In *AMW*, 2014.
[28] N. Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *J. Comb. Theory, Ser. B*, 41(1):92–114, 1986.
[29] B. L. Y. Chen, M. Grohe. The hardness of embedding grids and walls. In *WG*, pages 180–192, 2017.
[30] X. Zhang and J. V. den Bussche. On the power of SPARQL in expressing navigational queries. *The Computer Journal*, 58(11):2841–2851, 2015.

A APPENDIX

A.1 Proof of Lemma 4.4

LEMMA 4.4. *Let $k \geq 2$ and H be an undirected graph. Let (S, X) be a generalised t-graph with ctw$(S, X) \geq$ w$(\binom{k}{2})$. Then there is a generalised t-graph (B, X) such that*

(1) *if $t \in S$ and vars$(t) \subseteq X$, then $t \in B$.*
(2) *$(B, X) \to (S, X)$.*
(3) *H contains a clique of size k iff $(S, X) \to (B, X)$.*
(4) *(B, X) can be computed in time $f(k, |(S, X)|) \cdot |H|^{O(1)}$, where $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is a computable function.*

We devote this section to prove this lemma. Our proof is a simple modification of the main construction of [9] to handle distinguished elements.

We start with some definitions. For $k, \ell \geq 1$, the *$(k \times \ell)$-grid* is the undirected graph with vertex set $\{1, \ldots, k\} \times \{1, \ldots, \ell\}$ and an edge between (i, j) and (i', j') if $|i - i'| + |j - j'| = 1$. It is a known fact that the $(k \times k)$-grid has treewidth k (see e.g. [7]). We say that an undirected graph $H = (V, E)$ is a *minor* of $H' = (V', E')$ if there is a *minor map* from H to H', that is, a function γ mapping each vertex of H to a a non-empty set of vertices in H' such that (i) $\gamma(u)$ is connected for all $u \in V$, (ii) for all $u, v \in V$ with $u \neq v$, the sets $\gamma(u)$ and $\gamma(v)$ are disjoint, and (iii) for all edges $\{u, v\} \in E$, there is an edge $\{u', v'\} \in E'$ such that $u' \in \gamma(u)$ and $v' \in \gamma(v)$. We say that the minor map is *onto* if $\bigcup_{u \in V} \gamma(u) = V'$. Observe that if there is a minor map from H to H', and H' is connected, then there is a minor map of H onto H'.

Let $k \geq 2$, $H = (V, E)$ be an undirected graph, and (S, X) be a generalised t-graph with $\mathrm{ctw}(S, X) \geq \mathtt{w}(\binom{k}{2})$. From now on, we let $K := \binom{k}{2}$. Let (C, X) be the core of (S, X) and $G(C, X)$ be the Gaifman graph of (C, X). Suppose that $F_1, \ldots, F_r$ are the connected components of $G(C, X)$. As $\mathrm{tw}(G(C, X)) \geq \mathtt{w}(K)$, there is a connected component F_i with $\mathrm{tw}(F_i) \geq \mathtt{w}(K)$. Without loss of generality, we assume that $F_i = F_1$. By the Excluded Grid Theorem, since $\mathrm{tw}(F_1) \geq \mathtt{w}(K)$, it follows that the $(K \times K)$-grid is a minor of F_1, and hence $(k \times K)$-grid is a minor of F_1. Let γ be a minor map from the $(k \times K)$-grid onto F_1.

We fix a bijection ρ between $\{1, \ldots, K\}$ and all unordered pairs of elements of $\{1, \ldots, k\}$. For $p \in \{1, \ldots, K\}$, we shall abuse notation and write p instead of $\rho(p)$ and $i \in p$ instead of $i \in \rho(p)$, for $i \in \{1, \ldots, k\}$. We define the following set $\mathcal{V} \subseteq \mathbf{V}$ of variables: $?(v, e, i, p, ?a) \in \mathcal{V}$ iff $v \in V$, $e \in E$, $i \in \{1, \ldots, k\}$, $p \in \{1, \ldots, K\}$, $?a \in \gamma(i, p)$ and $v \in e \iff i \in p$.

We denote by $V(F_1) \subseteq \mathbf{V}$ the vertex set of F_1. Let $\Pi : (\mathcal{V} \cup \mathrm{vars}(C)) \to V(F_1)$ be the mapping such that $\Pi(?(v, e, i, p, ?a)) = ?a$, for all $?(v, e, i, p, ?a) \in \mathcal{V}$, and $\Pi(?x) = ?x$, for $?x \in \mathrm{vars}(C)$. We define

$$Tr := \{t \in (\mathbf{I} \cup \mathbf{V})^3 \mid \mathrm{vars}(t) \setminus X \subseteq \mathcal{V} \text{ and } \Pi(t) \in C\}.$$

We also define $Tr' \subseteq Tr$ as follows. For $t \in Tr$, we have $t \in Tr'$ iff (†) for all $?x, ?x' \in \mathrm{vars}(t) \setminus X \subseteq \mathcal{V}$.

(1) if $?x = ?(v, e, i, p, ?a)$ and $?x' = ?(v', e', i, p', ?a')$, then $v = v'$, and
(2) if $?x = ?(v, e, i, p, ?a)$ and $?x' = ?(v', e', i', p, ?a')$, then $e = e'$.

Let also $Tr_0 := \{t \in C \mid \mathrm{vars}(t) \setminus X \nsubseteq V(F_1)\}$. Then we define $B = Tr' \cup Tr_0$.

Note that Π is a homomorphism from B to C. Indeed, if $t \in Tr'$, then $t \in Tr$, and hence $\Pi(t) \in C$. If $t \in Tr_0$, then $\Pi(t) = t \in C$. Since $\Pi(?x) = ?x$, for all $?x \in X$, we have $(B, X) \to (C, X)$. Since $(C, X) \to (S, X)$, we have $(B, X) \to (S, X)$ and condition (2) in the lemma holds. For condition (1), let $t \in S$ such that $\mathrm{vars}(t) \subseteq X$. Since $(S, X) \to (C, X)$, then $t \in C$. In particular, $\mathrm{vars}(t) \setminus X = \emptyset \subseteq \mathcal{V}$ and, by definition of Π, we have $\Pi(t) = t \in C$. Hence $t \in Tr$. Since $\mathrm{vars}(t) \setminus X = \emptyset$, $t \in Tr'$ holds trivially. Then $t \in B$ as required. Condition (4) is immediate from the definition of B. It remains to verify condition (3). We can follow the same arguments as in [9].

Suppose that H contains a clique of size k and let $\{v_1, \ldots, v_k\}$ be such a clique. For $p \in K$ with $\rho(p) = \{i, j\}$, where $i, j \in \{1, \ldots, k\}$ and $i \neq j$, we let e_p be the edge from v_i to v_j. In this case we can define $h : \mathrm{vars}(C) \to \mathrm{vars}(B)$ such that $h(?x) = ?x$, if $?x \notin V(F_1)$, and $h(?a) = ?(v_i, e_p, i, p, ?a)$ if $?a \in V(F_1)$, for $i \in \{1, \ldots, k\}$ and $p \in \{1, \ldots, K\}$ with $\gamma(i, p) = ?a$. First note that $v_i \in e_p \iff i \in p$. Note that since v_i and e_p are determined by i and p, then for every $t \in C$, it is the case that $h(t) \in B$. As $h(?x) = ?x$, if $?x \in X$, we conclude that $(C, X) \to (B, X)$. Since $(S, X) \to (C, X)$, then $(S, X) \to (B, X)$ as required.

Conversely, suppose that $(S, X) \to (B, X)$. In particular, $(C, X) \to (B, X)$ via a homomorphism h. We claim that there is a homomorphism g from C to B with $g(?x) = ?x$, for all $?x \in X$ such that $\Pi \circ g$ is the identity mapping over $\mathrm{vars}(C)$. Indeed, let $s = \Pi \circ h$. Then s is a homomorphism witnessing $(C, X) \to (C, X)$. Since (C, X) is a core, then s must be a bijection and hence an isomorphism. It suffices to conisder $g = h \circ s^{-1}$. It follows that for all $i \in \{1, \ldots, k\}$, $p \in \{1, \ldots, K\}$, $?a$ such that $\gamma(i, p) = ?a$, $g(?a)$ is of the form

$$g(?a) = ?(v_{a?}, e_{?a}, i, p, ?a)$$

where $v_{?a} \in e_{?a} \iff i \in p$. By the consistency conditions (†) and the connectivity of F_1, it follows that (i) $v_{?a} = v_{?a'}$ and $e_{?a} = e_{?a'}$, whenever $?a, ?a' \in \gamma(i, p)$, (ii) $v_{?a} = v_{?a'}$, if $?a \in \gamma(i, p)$ and $?a' \in \gamma(i, p')$, and (iii) $e_{?a} = e_{?a'}$, if $?a \in \gamma(i, p)$ and $?a' \in \gamma(i', p)$. It follows that there are vertices $v_1, \ldots, v_k \in V$ and edges $e_1, \ldots, e_K$ such that whenever $?a \in \gamma(i, p)$ then $g(?a) = ?(v_i, e_p, i, p, ?a)$. By the conditions $v_i \in e_i \iff i \in p$, we have that $\{v_1, \ldots, v_k\}$ is a clique in H as required.