Constraint Satisfaction with Bounded Treewidth Revisited [⋆]

Marko Samer [a,1] and Stefan Szeider [b]

[a] *Department of Computer Science, TU Darmstadt, Germany*

[b] *Department of Computer Science, University of Durham, UK*

Abstract

We consider the constraint satisfaction problem (CSP) parameterized by the treewidth of primal, dual, and incidence graphs, combined with several other basic parameters such as domain size and arity. We determine all combinations of the considered parameters that admit fixed-parameter tractability.

Key words: Constraint satisfaction, parameterized complexity, treewidth

1 Introduction

An instance of the constraint satisfaction problem (CSP) consists of a set of variables that range over a domain of values together with a set of constraints that allow certain combinations of values for certain sets of variables. The question is whether one can instantiate the variables in such a way that all constraints are simultaneously satisfied; in that case the instance is called *consistent* or *satisfiable*. Constraint satisfaction provides a general framework which allows direct structure-preserving encodings of numerous problems that arise in practice.

Although constraint satisfaction is NP-complete in general, many efforts have been made to identify restricted problems that can be solved in polynomial time. Such restrictions can either limit the constraints used in the instance [6]

[⋆] Research supported by the EPSRC, project EP/E001394/1.
Email addresses: `samer@cs.tu-darmstadt.de` (Marko Samer), `stefan.szeider@durham.ac.uk` (Stefan Szeider).

[1] This research was carried out during the author's postdoc position at the University of Durham.

or limit the overall structure of the instance, i.e., how variables and constraints interact in the instance [8]. In this paper we focus on the latter form of restrictions which are also referred to as "structural restrictions." Structural restrictions are usually formulated in terms of certain graphs and hypergraphs that are associated with a CSP instance as described in the following.

The *primal graph* has the variables as its vertices; two variables are joined by an edge if they occur together in the scope of a constraint. The *dual graph* has the constraints as its vertices; two constraints are joined by an edge if their scopes have variables in common. The *incidence graph* is a bipartite graph and has both the variables and the constraints as its vertices; a variable and a constraint are joined by an edge if the variable occurs in the scope of the constraint. Finally, the *constraint hypergraph* is a hypergraph whose vertices are the variables and whose hyperedges are the constraint scopes.

Fundamental classes of tractable instances are obtained if the associated (hyper)graphs are *acyclic* with respect to certain notions of acyclicity. Acyclicity can be generalized by means of (hyper)graph decomposition techniques which give rise to "width" parameters that measure how far an instance deviates from being acyclic. Freuder [12] and Dechter and Pearl [9] observed that constraint satisfaction is polynomial-time solvable if

- the *treewidth of primal graphs*, tw,

is bounded by a constant. The graph parameter treewidth, introduced by Robertson and Seymour in their Graph Minors Project, has become a very popular object of study as many NP-hard graph problems are polynomial-time solvable for graphs of bounded treewidth; we define treewidth in Section 2.2. In subsequent years several further structural parameters have been considered, such as

- the *treewidth of dual graphs*, tw^d,
- the *treewidth of incidence graphs*, tw^*,

and various width parameters on constraint hypergraphs, including

- the *(generalized) hypertree-width*, $\mathsf{(g)hw}$, (Gottlob, Leone, and Scarcello [15]),
- the *spread-cut-width*, scw, (Cohen, Jeavons, and Gyssens [7]), and
- the *fractional hypertree-width*, fhw, (Grohe and Marx [18]).

Considering CSP instances where the width parameter under consideration is bounded by some fixed integer k gives rise to a class W_k of tractable instances. The larger k gets, the larger is the resulting tractable class W_k. However, for getting larger and larger tractable classes one has to pay by longer running times. A fundamental question is the *trade-off between generality and performance.* A typical time complexity of algorithms known from the literature are of the form

$$\mathcal{O}(\|I\|^{f(k)}) \tag{1}$$

for instances I belonging to the class W_k; here $\|I\|$ denotes the input size of I and $f(k)$ denotes a slowly growing function. Such a running time is polynomial when k is considered as a constant. However, since k appears in the exponent, such algorithms become impractical—even if k is small—when large instances are considered. It is significantly better if instances I of the class W_k can be solved in time

$$\mathcal{O}(f(k)\,\|I\|^c) \tag{2}$$

where f is an arbitrary (possibly exponential) computable function and c is a constant independent of k and I. In that case the order of the polynomial does not depend on k, and so considering larger and larger classes does not increase the order of the polynomial. Thus, it is of interest to classify the trade-off between generality and performance of a width parameter under consideration: whether the parameter allows algorithms of type (1) or of type (2).

1.1 Parameterized Complexity

The framework of *parameterized complexity* provides the adequate concepts and tools for studying the above question. Parameterized complexity was initiated by Downey and Fellows in the late 1980s and has become an important branch of algorithm design and analysis; hundreds of research papers have been published in that area (see the references in [10,11,21]). It has turned out that the distinction between tractability of type (1) and tractability of type (2) is a robust indication of problem hardness.

A *fixed parameter algorithm* is an algorithm that achieves a running time of type (2). A parameterized problem is *fixed-parameter tractable* if it can be solved by a fixed-parameter algorithm. FPT denotes the class of all fixed-parameter tractable decision problems.

Parameterized complexity offers a *completeness theory*, similar to the theory of NP-completeness, that allows the accumulation of strong theoretical evidence that a parameterized problem is *not* fixed-parameter tractable. This completeness theory is based on the *weft hierarchy* of complexity classes $\mathrm{W}[1], \mathrm{W}[2], \dots, \mathrm{W}[\mathrm{P}]$. Each class is the equivalence class of certain parameterized satisfiability problems under *fpt-reductions* (for instance, the canonical W[1]-complete problem asks whether a given 3SAT instance can be satisfied by setting at most k variables to *true*). Let Π and Π' be two parameterized problems. An *fpt-reduction R from Π to Π'* is a many-to-one transformation from Π to Π', such that

(i) $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi'$ with $k' \leq g(k)$ for a fixed computable function g and

(ii) R is of complexity $\mathcal{O}(f(k)\,\|I\|^c)$ for a computable function f and a constant c.

The class XP consists of parameterized problems which can be solved in polynomial time if the parameter is considered as a constant. The above classes form the chain

$$\mathrm{FPT} \subseteq \mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots \subseteq \mathrm{W}[\mathrm{P}] \subseteq \mathrm{XP}$$

where all inclusions are believed to be proper. A parameterized analog of Cook's Theorem [10] as well as the Exponential Time Hypothesis [11,19] give strong evidence to assume that FPT $\neq$ W[1]. It is known that FPT $\neq$ XP [10]. Although XP contains problems which are very unlikely to be fixed-parameter tractable, it is often a significant improvement to show that a problem belongs to this class, in contrast to, e.g., k-SAT which is NP-complete for every constant $k \geq 3$.

The following parameterized clique-problem and independent set-problem are W[1]-complete [10]; these problems are the basis for the hardness results considered in the sequel.

Clique
Instance: A graph G and a non-negative integer k.
Parameter: k.
Question: Does G contain a clique on k vertices?

Independent Set
Instance: A graph G and a non-negative integer k.
Parameter: k.
Question: Does G contain an independent set on k vertices?

1.2 Parameterized Constraint Satisfaction

We consider any computable function p that assigns to a CSP instance I a non-negative integer $p(I)$ as a *CSP parameter*; CSP instances are formally defined in Section 2.1. For CSP parameters $p_1, \dots, p_r$ we consider the following generic parameterized problem:

CSP$(p_1, \dots, p_r)$
Instance: A CSP instance I and non-negative integers $k_1, \dots, k_r$ with $p_1(I) \leq k_1, \dots, p_r(I) \leq k_r$.
Parameters: $k_1, \dots, k_r$.
Question: Is I consistent?

Slightly abusing notation, we will also write CSP(S) for a set S of parameters, assuming an arbitrary but fixed ordering of the parameters in S. We write $\mathrm{CSP}_{\mathrm{boole}}(S)$ to denote CSP(S) with the *Boolean domain* $\{0,1\}$, and $\mathrm{CSP}_{\mathrm{bin}}(S)$ to denote CSP(S) where all constraints have arity at most 2.

Note that we formulate the problem $\mathrm{CSP}(p_1,\dots,p_r)$ as a "promise problem" in the sense that for solving the problem we do not need to verify the assumption $p_1(I) \leq k_1,\dots,p_r(I) \leq k_r$. However, unless otherwise stated, for all cases considered in the sequel where $\mathrm{CSP}(p_1,\dots,p_r)$ is fixed-parameter tractable, also the verification of the assumption $p_1(I) \leq k_1,\dots,p_r(I) \leq k_r$ is fixed-parameter tractable. For a CSP instance I we have the following basic parameters:

- the number of variables, vars,
- the number of values, dom,
- the number of constraints, cons,
- the largest size of a constraint scope, arity,
- the largest size of a relation, dep,
- the largest number of occurrences of a variable, deg,
- the largest overlap between two constraint scopes, ovl,
- the largest difference between two constraint scopes, diff.

Gottlob, Scarcello, and Sideri [17] have determined the parameterized complexity of constraint satisfaction with respect to the treewidth of primal graphs: $\mathrm{CSP}(\mathsf{tw},\mathsf{dom})$ is fixed-parameter tractable and $\mathrm{CSP}(\mathsf{tw})$ is W[1]-hard. The parameterized complexity of constraint satisfaction with respect to other structural parameters like treewidth of dual graphs, treewidth of incidence graphs, and the more general width parameters defined in terms of constraint hypergraphs remained open. In this paper we determine exactly those combinations of parameters from tw, tw^d, tw^*, vars, dom, cons, arity, dep, deg, ovl, and diff that render constraint satisfaction fixed-parameter tractable.

The following concept allows us to establish the classification by considering only a few border case. Let S and $S' = \{p'_1, p'_2, \dots, p'_{r'}\}$ be two finite sets of CSP parameters. S *dominates* S' if for every $p \in S$ there exists an r'-ary computable function f that is monotonically increasing in each argument such that for every CSP instance I we have $p(I) \leq f(p'_1(I), p'_2(I), \dots, p'_{r'}(I))$. See Lemma 2 for examples that illustrate this notion (if S or S' is a singleton, we omit the braces to improve readability). It is easy to see that whenever S dominates S', then fixed-parameter tractability of CSP(S) implies fixed-parameter tractability of CSP(S'), and W[1]-hardness of CSP(S') implies W[1]-hardness of CSP(S) (see Lemma 1).

We obtain the following classification result (see also the diagram in Figure 1 and the discussion in Section 3).

Theorem 1 (Classification Theorem) *Let* $S \subseteq \{\mathsf{tw}, \mathsf{tw}^d, \mathsf{tw}^*, \mathsf{vars}, \mathsf{dom}, \mathsf{cons}, \mathsf{arity}, \mathsf{dep}, \mathsf{deg}, \mathsf{ovl}, \mathsf{diff}\}$.

(1) If $\{\mathsf{tw}^*, \mathsf{dep}\}$, $\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}\}$, *or* $\{\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}\}$ *dominates* S, *then* $\mathrm{CSP}(S)$ *is fixed-parameter tractable.*
(2) Otherwise, if neither of them dominates S, *then, unless* $\mathrm{FPT} = \mathrm{W}[1]$,
  (a) $\mathrm{CSP}(S)$ *is not fixed-parameter tractable.*
  (b) if $\mathsf{dom} \in S$, *then even* $\mathrm{CSP}_{\mathrm{boole}}(S)$ *is not fixed-parameter tractable.*
  (c) if $\mathsf{arity} \in S$, *then even* $\mathrm{CSP}_{\mathrm{bin}}(S)$ *is not fixed-parameter tractable.*

The complexity theoretic assumption $\mathrm{FPT} \neq \mathrm{W}[1]$ is discussed in Section 1.1.

The notion of domination allows us to extend the W[1]-hardness results of the Classification Theorem to all parameters that are more general than the treewidth of incidence graphs. In particular, we obtain the following corollary to Theorem 1.

Corollary 1 *The problems* $\mathrm{CSP}(p, \mathsf{dom})$ *and* $\mathrm{CSP}_{\mathrm{boole}}(p)$ *are* W[1]*-hard if* p *is any of the parameters treewidth of incidence graphs, hypertree-width, generalized hypertree-width, spread-cut-width, and fractional hypertree-width.*

Recently, Gottlob et al. [14] have shown that the problem of deciding whether a given hypergraph has (generalized) hypertree-width at most k, is W[2]-hard. Note that this result does not imply W[1]-hardness of $\mathrm{CSP}(\mathsf{hw}, \mathsf{dom})$ (respectively $\mathrm{CSP}(\mathsf{ghw}, \mathsf{dom})$), since it is possible to design algorithms for CSP instances of bounded (generalized) hypertree-width that avoid the decomposition step. Chen and Dalmau [5] have recently proposed such an algorithm, which, however, is not a fixed-parameter algorithm.

Our results indicate a somewhat surprising difference between Boolean constraint satisfaction and propositional satisfiability (SAT). A SAT instance is a set of clauses, representing a propositional formula in conjunctive normal form. The question is whether the instance is satisfiable. Primal, dual, and incidence graphs and the corresponding treewidth parameters tw, tw^d, and tw^* can be defined for SAT similarly as for constraint satisfaction [27], as well as the parameterized decision problem $\mathrm{SAT}(p)$ for a parameter p. In contrast to the W[1]-hardness of $\mathrm{CSP}_{\mathrm{boole}}(\mathsf{tw}^*)$, as established in Corollary 1, the problem $\mathrm{SAT}(\mathsf{tw}^*)$ is fixed-parameter tractable. This holds also true for $\mathrm{SAT}(\mathsf{tw}^d)$ and $\mathrm{SAT}(\mathsf{tw})$ since tw^* dominates tw^d and tw. Szeider [27] has shown the fixed-parameter tractability of $\mathrm{SAT}(\mathsf{tw}^*)$ by using a general result for Monadic Second Order (MSO) logic on graphs; Samer and Szeider [25] have developed an efficient dynamic programming algorithm for this problem. We refer the interested reader to a recent survey chapter [26] on fixed-parameter tractability and SAT.

2 Preliminaries

2.1 Constraint Satisfaction

Formally, a CSP instance I is a triple $(V, D, \mathcal{C})$, where V is a finite set of *variables*, D is a finite set of *domain values*, and $\mathcal{C}$ is a finite set of *constraints*. Each constraint in $\mathcal{C}$ is a pair (S, R), where S, the *constraint scope*, is a non-empty sequence of distinct variables of V, and R, the *constraint relation*, is a relation over D whose arity matches the length of S; a relation is considered as a set of tuples. We assume w.l.o.g. that every variable occurs in at least one constraint scope and every domain element occurs in at least one constraint relation. We write $\mathit{var}(C)$ for the set of variables that occur in the scope of constraint C, $\mathit{rel}(C)$ for the relation of C, and $\mathit{con}(x)$ for the set of constraints that contain variable x in their scopes. Moreover, for a set $\mathcal{C}$ of constraints, we set $\mathit{var}(\mathcal{C}) = \bigcup_{C \in \mathcal{C}} \mathit{var}(C)$.

An *assignment* is a mapping $\tau : X \to D$ defined on some set X of variables. Let $C = ((x_1, \ldots, x_n), R)$ be a constraint and $\tau : X \to D$ an assignment. We define

$$C[\tau] = \{\, (d_1, \ldots, d_n) \in R : x_i \notin X \text{ or } \tau(x_i) = d_i,\ 1 \le i \le n \,\}.$$

Thus, $C[\tau]$ contains those tuples of R that do not disagree with τ at some position. An assignment $\tau : X \to D$ is *consistent* with a constraint C if $C[\tau] \neq \emptyset$. An assignment $\tau : X \to D$ *satisfies* a constraint C if $\mathit{var}(C) \subseteq X$ and τ is consistent with C. An assignment satisfies a CSP instance I if it satisfies all constraints of I. The instance I is *consistent* (or *satisfiable*) if it is satisfied by some assignment. The *constraint satisfaction problem* is the problem of deciding whether a given CSP instance is consistent (resp. satisfiable).

A constraint $C = ((x_1, \ldots, x_n), R)$ is the *projection* of constraint $C' = (S', R')$ to $V \subset \mathit{var}(C')$ if $V = \{x_1, \ldots, x_n\}$ and R consists of all tuples $(\tau(x_1), \ldots, \tau(x_n))$ for assignments τ that are consistent with C'. If C is a projection of C', we say that C is obtained from C' by *projecting out* all variables in $\mathit{var}(C') \setminus \mathit{var}(C)$. A constraint $C = ((x_1, \ldots, x_n), R)$ is the *join* of constraints $C_1, \ldots, C_r$ if $\mathit{var}(C) = \bigcup_{i=1}^{r} \mathit{var}(C_i)$ and if R consists of all tuples $(\tau(x_1), \ldots, \tau(x_n))$ for assignments τ that are consistent with C_i for all $1 \le i \le r$ (cf. Abiteboul et al. [1]).

Let G be a graph, T a rooted tree, and χ a labeling of the vertices of T by sets of vertices of G. We refer to the vertices of T as “nodes” to avoid confusion with the vertices of G, and we call the sets $\chi(t)$ “bags.” For each node t of T we denote by T_t the subtree of T rooted at t. The pair (T, χ) is a *tree decomposition* of G if the following three conditions hold:

(1) For every vertex v of G there exists a node t of T such that $v \in \chi(t)$.
(2) For every edge vw of G there exists a node t of T such that $v, w \in \chi(t)$.
(3) For any three nodes t_1, t_2, t_3 of T, if t_2 lies on the unique path from t_1 to t_3, then $\chi(t_1) \cap \chi(t_3) \subseteq \chi(t_2)$ (“Connectedness Condition”).

The *width* of a tree decomposition (T, χ) is defined as the maximum $|\chi(t)| - 1$ over all nodes t of T. The *treewidth* $\mathrm{tw}(G)$ of a graph G is the minimum width over all its tree decompositions.

As shown by Bodlaender [2], there exists for every fixed k a linear time algorithm that checks whether a given graph has treewidth at most k and, if so, outputs a tree decomposition of minimum width. Bodlaender’s algorithm does not seem feasible to implement [4]. However, there are several other known fixed-parameter algorithms that are feasible. For example, Reed’s algorithm [23] runs in time $\mathcal{O}(|V| \log |V|)$ for any fixed k and decides either that the treewidth of the given graph $G = (V, E)$ exceeds k, or outputs a tree decomposition of width at most $4k$. The algorithm produces tree decompositions with $\mathcal{O}(|V|)$ many nodes.

Let (T, χ) be a tree decomposition of a graph G and let r be a node of T. The triple (T, χ, r) is a *nice tree decomposition* of G if the following three conditions hold (here we consider T as a tree rooted at r):

(1) Every node of T has at most two children.
(2) If a node t of T has two children t_1 and t_2, then $\chi(t) = \chi(t_1) = \chi(t_2)$; in that case we call t a *join node*.
(3) If a node t of T has exactly one child t', then exactly one of the following prevails:
    (a) $|\chi(t)| = |\chi(t')| + 1$ and $\chi(t') \subset \chi(t)$; in that case we call t an *introduce node*.
    (b) $|\chi(t)| = |\chi(t')| - 1$ and $\chi(t) \subset \chi(t')$; in that case we call t a *forget node*.

It is well known (and easy to see) that for every fixed k, given a tree decomposition of a graph $G = (V, E)$ of width at most k and with $\mathcal{O}(|V|)$ nodes, one can construct in linear time a nice tree decomposition of G with $\mathcal{O}(|V|)$ nodes and width at most k [4].

3 The Domination Lattice

Lemma 1 *Let S and S' be two sets of CSP parameters such that S dominates S'. Then there is an fpt-reduction from* $\mathrm{CSP}(S')$ *to* $\mathrm{CSP}(S)$*. In particular, fixed-parameter tractability of* $\mathrm{CSP}(S)$ *implies fixed-parameter tractability of* $\mathrm{CSP}(S')$*, and* $\mathrm{W}[1]$*-hardness of* $\mathrm{CSP}(S')$ *implies* $\mathrm{W}[1]$*-hardness of* $\mathrm{CSP}(S)$*.*

PROOF. Let $S = \{p_1, \ldots, p_r\}$ and $S' = \{p'_1, \ldots, p'_{r'}\}$ and assume that S dominates S'. By definition, for every $i = 1, \ldots, r$ there exists an r'-ary computable function f_i that is monotonically increasing in each argument such that for every CSP instance I we have $p_i(I) \leq f_i(p'_1(I), p'_2(I), \ldots, p'_{r'}(I))$. Consider an instance $(I, k'_1, \ldots, k'_{r'})$ of $\mathrm{CSP}(S')$; i.e., we have $p'_i(I) \leq k'_i$ for all $1 \leq i \leq r'$. We put $k_i = f_i(k'_1, k'_2, \ldots, k'_{r'})$ for all $1 \leq i \leq r$. Since f_i is monotonically increasing in each argument, we have $p_i(I) \leq f_i(p'_1(I), p'_2(I), \ldots, p'_{r'}(I)) \leq f_i(k'_1, k'_2, \ldots, k'_{r'}) = k_i$. Hence $(I, k_1, \ldots, k_r)$ is an instance of $\mathrm{CSP}(S)$. Whence we have indeed an fpt-reduction from $\mathrm{CSP}(S')$ to $\mathrm{CSP}(S)$. The second part of the lemma is a direct consequence of the first part. $\square$

Next we give a more formal definition of the basic CSP parameters introduced in Section 1.2. For a CSP instance $I = (V, D, \mathcal{C})$ the parameters are defined as follows:

- $\mathsf{vars}(I) = |V|$
- $\mathsf{dom}(I) = |D|$
- $\mathsf{cons}(I) = |\mathcal{C}|$
- $\mathsf{arity}(I) = \max_{C \in \mathcal{C}} |var(C)|$
- $\mathsf{dep}(I) = \max_{C \in \mathcal{C}} |rel(C)|$
- $\mathsf{deg}(I) = \max_{x \in V} |con(x)|$
- $\mathsf{ovl}(I) = \max_{C, C' \in \mathcal{C}, C \neq C'} |var(C) \cap var(C')|$
- $\mathsf{diff}(I) = \max_{C, C' \in \mathcal{C}} |var(C) \setminus var(C')|$

The following lemma provides the basis for deriving all domination relations between the sets of parameters we consider in the sequel of this paper.

Lemma 2

(1) If $S \subseteq S'$, then S dominates S'.
(2) tw *dominates* vars.
(3) tw *dominates* $\{\mathsf{tw}^*, \mathsf{arity}\}$.
(4) tw^d *dominates* cons.
(5) tw^d *dominates* $\{\mathsf{tw}^*, \mathsf{deg}\}$.
(6) tw^* *dominates* tw.
(7) tw^* *dominates* tw^d.
(8) vars *dominates* $\{\mathsf{cons}, \mathsf{arity}\}$.
(9) dom *dominates* $\{\mathsf{cons}, \mathsf{arity}, \mathsf{dep}\}$.
(10) cons *dominates* $\{\mathsf{vars}, \mathsf{deg}\}$.
(11) arity *dominates* tw.
(12) dep *dominates* $\{\mathsf{dom}, \mathsf{arity}\}$.
(13) deg *dominates* tw^d.
(14) ovl *dominates* arity.
(15) diff *dominates* arity.

PROOF. Part 1 is obvious. Parts 2 and 4 follow from the fact that there is always a trivial tree decomposition of the primal graph and of the dual graph of I of width $\mathsf{vars}(I)-1$ and $\mathsf{cons}(I)-1$, respectively. Thus, $\mathsf{tw}(I) \leq \mathsf{vars}(I)-1$ and $\mathsf{tw}^d(I) \leq \mathsf{cons}(I)-1$. Part 3 follows from the fact that every tree decomposition of the incidence graph can be transformed into a tree decomposition of the primal graph by replacing each constraint C in the bags by $var(C)$. Thus, $\mathsf{tw}(I) \leq \mathsf{tw}^*(I)(\mathsf{arity}(I)-1)$ as observed by Kolaitis and Vardi [20]. A symmetric argument applies to part 5, i.e., every tree decomposition of the incidence graph can be transformed into a tree decomposition of the dual graph by replacing each variable x in the bags by $con(x)$. Thus, $\mathsf{tw}^d(I) \leq \mathsf{tw}^*(I)(\mathsf{deg}(I)-1)$. Part 6 follows from the inequality $\mathsf{tw}^*(I) \leq \mathsf{tw}(I)+1$ shown by Kolaitis and Vardi [20]; a symmetric argument gives $\mathsf{tw}^*(I) \leq \mathsf{tw}^d(I)+1$, hence part 7 holds as well. Parts 8, 9, 10, and 12 follow from the obvious inequalities $\mathsf{vars}(I) \leq \mathsf{cons}(I) \cdot \mathsf{arity}(I)$, $\mathsf{dom}(I) \leq \mathsf{cons}(I) \cdot \mathsf{arity}(I) \cdot \mathsf{dep}(I)$, $\mathsf{cons}(I) \leq \mathsf{vars}(I) \cdot \mathsf{deg}(I)$, and $\mathsf{dep}(I) \leq \mathsf{dom}(I)^{\mathsf{arity}(I)}$, respectively. (Recall for parts 8 and 9 that we assume w.l.o.g. that every variable occurs in at least one scope and every domain element occurs in at least one relation.) Part 11 follows from the fact that a constraint of arity r yields a clique on r vertices in the primal graph; it is well known that if a graph G contains a clique with r vertices, then $\mathrm{tw}(G) \geq r-1$ [3]. Thus, $\mathsf{arity}(I) \leq \mathsf{tw}(I)+1$. A symmetric argument applies to part 13, i.e., a variable of degree r yields a clique on r vertices in the dual graph. Thus, $\mathsf{deg}(I) \leq \mathsf{tw}^d(I)+1$. Finally, parts 14 and 15 follow from the obvious inequalities $\mathsf{ovl}(I) \leq \mathsf{arity}(I)$ and $\mathsf{diff}(I) \leq \mathsf{arity}(I)$, respectively. □

Note that parts 2, 4, and 6–15 in the above lemma are *strict* in the sense that p dominates q but q does not dominate p.

Based on Lemmas 1 and 2, we obtain the following corollary to the Classification Theorem by simply going through all subsets of parameters and checking whether they are dominated by $\{\mathsf{tw}^*, \mathsf{dep}\}$, $\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}\}$ or $\{\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}\}$.

Corollary 2 *Let* $S \subseteq \{\mathsf{tw}, \mathsf{tw}^d, \mathsf{tw}^*, \mathsf{vars}, \mathsf{dom}, \mathsf{cons}, \mathsf{arity}, \mathsf{dep}, \mathsf{deg}, \mathsf{ovl}, \mathsf{diff}\}$. *Then* CSP($S$) *is fixed-parameter tractable if* S *contains at least one of the following 14 sets as subset:*

$$
\begin{array}{lll}
\{\mathsf{tw}, \mathsf{dom}\}, & \{\mathsf{tw}, \mathsf{dep}\}, & \{\mathsf{vars}, \mathsf{dom}\}, \\
\{\mathsf{tw}^d, \mathsf{dom}, \mathsf{arity}\}, & \{\mathsf{tw}^d, \mathsf{dep}\}, & \{\mathsf{dom}, \mathsf{cons}, \mathsf{arity}\}, \\
\{\mathsf{tw}^d, \mathsf{dom}, \mathsf{diff}\}, & \{\mathsf{tw}^*, \mathsf{dep}\}, & \{\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}\}, \\
\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{arity}\}, & \{\mathsf{vars}, \mathsf{dep}\}, & \{\mathsf{dom}, \mathsf{cons}, \mathsf{diff}\}, \\
\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}\}, & \{\mathsf{cons}, \mathsf{dep}\}. &
\end{array}
$$

Otherwise, if none of them is a subset of S, CSP(S) *is not fixed-parameter tractable unless* FPT = W[1].

Let us now consider the sets S of parameters that are not dominated by $\{\mathsf{tw}^*, \mathsf{dep}\}$, $\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}\}$, or $\{\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}\}$, and let us assume that CSP(S) is W[1]-hard for these sets as indicated by the Classification Theorem. Thus by Lemma 2(a), we know that for each of these sets S, if $S' \subseteq S$, then also CSP(S') is W[1]-hard. Consequently, it suffices to consider those sets of parameters that are not subset of another set. This yields a characterization dual to Corollary 2. Thus, together with the sets listed in Corollary 2, we obtain our domination lattice in Figure 1. Note that if two sets are *domination equivalent* (that is, if they dominate each other and thus trivially have the same parameterized complexity), we consider only one of them in the lattice. For example, $\{\mathsf{tw}, \mathsf{dom}\}$ and $\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{arity}\}$ are domination equivalent.

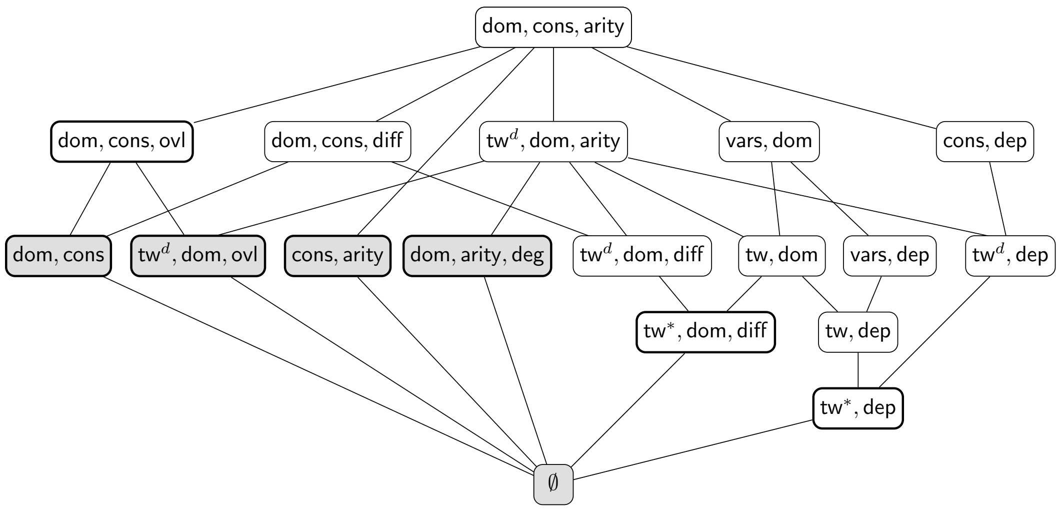

Fig. 1. Domination lattice

The domination lattice shows the relationships among the resulting 17 sets: A set S dominates a set S' if and only if there is a path running upwards from S to S'; in fact, it can be easily shown that whenever one of the 17 sets dominates another, it strictly dominates the other set. The sets S for which CSP(S) is fixed-parameter tractable according to the Classification Theorem are indicated in the lattice by boxes with rounded corners.

In order to prove the Classification Theorem, we will need the following proposition, which can be easily read off from the domination lattice.

Proposition 1 *Let* $S \subseteq \{\mathsf{tw}, \mathsf{tw}^d, \mathsf{tw}^*, \mathsf{vars}, \mathsf{dom}, \mathsf{cons}, \mathsf{arity}, \mathsf{dep}, \mathsf{deg}, \mathsf{ovl}, \mathsf{diff}\}$. *If* S *is not dominated by* $\{\mathsf{tw}^*, \mathsf{dep}\}$, $\{\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}\}$, *or* $\{\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}\}$, *then* S *dominates* $\{\mathsf{tw}^d, \mathsf{dom}, \mathsf{ovl}\}$, $\{\mathsf{cons}, \mathsf{arity}\}$, $\{\mathsf{dom}, \mathsf{cons}\}$, *or* $\{\mathsf{dom}, \mathsf{arity}, \mathsf{deg}\}$.

In view of Proposition 1, Theorem 1 is established if we show (i) fixed-parameter tractability of CSP($\mathsf{tw}^*, \mathsf{dep}$), CSP($\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}$), and CSP($\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}$), as well as (ii) W[1]-hardness of CSP($\mathsf{tw}^d, \mathsf{dom}, \mathsf{ovl}$), CSP($\mathsf{cons}, \mathsf{arity}$), CSP($\mathsf{dom}, \mathsf{cons}$), and CSP($\mathsf{dom}, \mathsf{arity}, \mathsf{deg}$).

Let us start with the fixed-parameter tractability of CSP($\mathsf{tw}^*, \mathsf{dep}$). To this aim, we will prove a series of auxiliary results that are of independent interest.

Let $I = (V, D, \mathcal{C})$ be a CSP instance, $x \in V$, and $C_1, \dots, C_r \in \mathcal{C}$ the constraints with $v \in var(C_i)$ for all $1 \leq i \leq r$, $r > 3$. Now let us construct a CSP instance $I' = (V', D, \mathcal{C}')$ as follows: We take a new variable $x' \notin V$ and put $V' = V \cup \{x'\}$. We take the new constraint $C_{x=x'} = ((x, x'), =_D)$, where $=_D$ denotes the equality relation $\{\, (d, d) : d \in D \,\}$. For $i \in \{1, 2\}$, let $C_i^{x'}$ denote the constraint obtained from C_i by replacing x by x' in the scope of C_i. We put $\mathcal{C}' = (\mathcal{C} \setminus \{C_1, C_2\}) \cup \{C_1^{x'}, C_2^{x'}, C_{x=x'}\}$. Evidently, I and I' are either both consistent or both inconsistent. Now variable x occurs in the scopes of $r - 1$ constraints of I'. By repeating this construction $r - 3$ times, we finally are left with a CSP instance where x occurs in the scopes of at most three constraints. Further, we can apply a similar construction with respect to other variables, obtaining an instance I^* where all variables occur in the scopes of at most three constraints. We say that I^* is obtained from I by *splitting*. In particular, denoting by $d(x)$ the number of constraints of I containing variable x in their scopes, we obtain I^* by repeating the above procedure $\sum_{x \in V} \max(0, d(x) - 3)$ times. The following lemma summarizes this construction:

Lemma 3 *Given a CSP instance I, we can obtain in polynomial time a CSP instance I^* such that the following holds:*

(1) I is consistent if and only if I^* is consistent.
(2) Each variable of I^* occurs in the scopes of at most three constraints of I^*.

Note that the splitting procedure described above does not yield a unique instance I^* as the construction depends on the chosen ordering of the constraints $C_1, \dots, C_r$ in each splitting step. The following result indicates that one needs to choose the orderings carefully to avoid and unbounded increase of incidence treewidth.

Proposition 2 *There are CSP instances of constant incidence treewidth from which one can obtain by splitting instances of arbitrarily high incidence treewidth.*

PROOF. We consider a family of CSP instances I_n consisting of variables x_i for $1 \leq i \leq n$ and variables $y_{i,j}$ for $0 \leq i \leq n$ and $1 \leq j \leq 2n$. Further, I_n has constraints $C_{i,j}$ for $0 \leq i \leq n$ and $0 \leq j \leq 2n$, with $x_i \in var(C_{i',j'})$ if and only if $i \in \{i' - 1, i'\}$, and $y_{i,j} \in var(C_{i',j'})$ if and only if $i = i'$ and $j \in \{j' - 1, j'\}$. It is not difficult to verify that $\mathsf{tw}^*(I_n) = 3$; this follows for example from

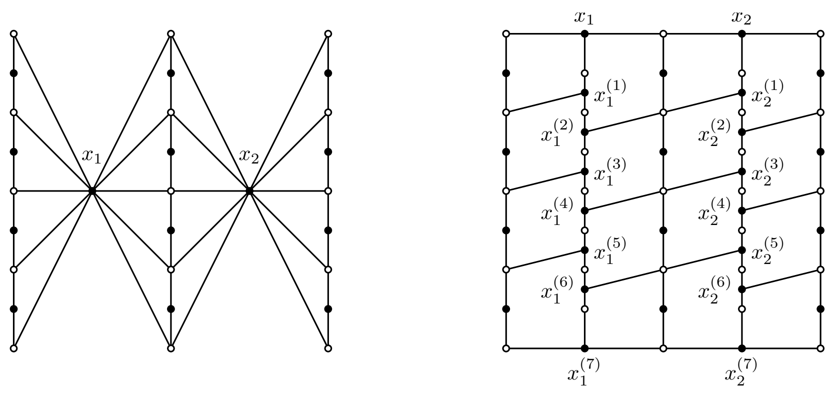

Fig. 2. Incidence graphs of I_2 and I_2^*, respectively

the observation that three cops can search the incidence graph of I_n in the Seymour-Thomas search game [3]. For an example see the left hand side of Figure 2 for the case $n = 2$.

We obtain from I_n the instance I_n^* by splitting, performing the splitting in a particular way, resulting in an incidence graph as depicted at the right hand side of Figure 2 for the case $n = 2$. It is straightforward to formalize this construction for arbitrary n. Evidently, the incidence graph of I_n^* contains a $(2n + 1) \times (2n + 1)$ grid as a minor (a graph G is a *minor* of a graph H if G can be obtained from a subgraph of H by contraction of edges; we contract an edge uv by replacing the vertices u and v by a new vertex x that is adjacent to all the vertices that were adjacent to u or v). Using the well-known facts that the treewidth of an $r \times r$ grid equals r, and that the treewidth of a graph is at least as large as the treewidth of any of its minors (see, e.g., Bodlaender [3]), we conclude that $\mathsf{tw}^*(I_n^*) \geq 2n + 1$. $\square$

Next we show that by carefully choosing the ordering we can always find a splitting such that the incidence treewidth increases at most by one.

Lemma 4 *Given a CSP instance I together with a tree decomposition of width k of the incidence graph of I. By splitting I we can obtain I^* with incidence treewidth at most $k + 1$ in polynomial time. Moreover, within the same time bounds we can construct a tree decomposition of the incidence graph of I^* of width at most $k + 1$ such that each bag contains at most $k + 1$ variables.*

PROOF. Let us call a nice tree decomposition (T, χ, r) of the incidence graph of a CSP instance $I = (V, D, \mathcal{C})$ to be *k-special* if the following conditions hold:

(1) $\chi(r) = \emptyset$.
(2) $|\chi(t)| \leq k + 2$ and $|\chi(t) \cap V| \leq k + 1$ hold for every node t of T.

(3) If $|\chi(t)| = k+2$ for a node t of T, then t is an introduce node that is not the child of a forget node where a variable is forgotten that occurs in the scopes of more than three constraints.

Now, given an arbitrary tree decomposition of width k of the incidence graph of I, we can convert it in polynomial time into a nice tree decomposition of the same width. Moreover, by adding new forget nodes on top of the root, we can easily obtain a nice tree decomposition (T, χ, r) of width k where $\chi(r) = \emptyset$. Evidently, (T, χ, r) is k-special.

If each variable of I occurs in the scopes of at most three constraints, then we have nothing to show. Hence assume that some variable x occurs in the scopes of more than three constraints. Let $S = \{C_1, \ldots, C_s\}$ be the set of constraints of I having x in their scopes. Since $\chi(r) = \emptyset$, there must be a unique forget node t_x where x is forgotten, i.e., $x \notin \chi(t_x)$ but $x \in \chi(t'_x)$ for the single child t'_x of t_x. Note that all constraints in S must occur in bags of nodes of the subtree rooted at t'_x since otherwise an edge of the incidence graph (between x and a constraint in S) would not be covered by the tree decomposition. We distinguish two cases:

Case 1: $|S \cap \chi(t_x)| \leq s-2$.

Hence at least two constraints from S are already forgotten at t_x. Let t be the lowest node in T where exactly two constraints in S, say C_1 and C_2, are forgotten. Note that t must be either a forget node at which C_1 or C_2 is forgotten, or a join node. Since $s > 3$, t cannot be the root, hence it has a parent t'.

We form a new CSP instance I' by introducing a new variable x' and replacing the constraints C_1, C_2 by the constraints $C_1^{x'}, C_2^{x'}, C_{x=x'}$ as described above. To get a nice tree decomposition (T', χ', r) of the incidence graph of I' we modify (T, χ, r) as follows: First we replace in the bags of t and all nodes below t the variable x with x' and constraint C_i with $C_i^{x'}$, $i \in \{1,2\}$. Second, we replace the edge tt' of T with the path $tt_1t_2t_3t_4t'$, where $t_1, \ldots, t_4$ are new nodes. The bags of the new nodes are defined as follows:

$$\begin{aligned}
\chi'(t_1) &= \chi'(t) \cup \{C_{x=x'}\} && \text{(introduce } C_{x=x'}\text{)}\\
\chi'(t_2) &= \chi'(t_1) \setminus \{x'\} && \text{(forget } x'\text{)}\\
\chi'(t_3) &= \chi'(t_2) \cup \{x\} && \text{(introduce } x\text{)}\\
\chi'(t_4) &= \chi'(t_3) \setminus \{C_{x=x'}\} && \text{(forget } C_{x=x'}\text{)}
\end{aligned}$$

It can be easily verified that this construction gives indeed a nice tree decomposition (T', χ', r) of I'. Since (T, χ, r) is k-special and t is either a forget node or a join node, $|\chi(t)| = |\chi'(t)| \leq k+1$. Consequently, for $i \in \{2,4\}$ we have

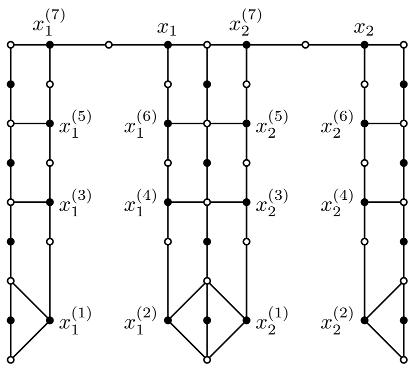

Fig. 3. Incidence graph of I_2^* obtained by splitting I_2 from Figure 2 using Lemma 4

$|\chi'(t_i)| \leq k+1$; for $i \in \{1,3\}$ we have $|\chi'(t_i)| \leq k+2$ and $|\chi'(t_i) \cap V| \leq k+1$, however, t_i is a child of a forget node where either a constraint is forgotten or a variable is forgotten that occurs in the scopes of exactly three constraints. Hence, (T', χ', r) is k-special.

Case 2: $|S \cap \chi(t_x)| \geq s-1$.

Hence at most one constraint from S, say C_1, is already forgotten at t_x. As above we form a new CSP instance I' by introducing a new variable x' and replacing the constraints C_1, C_2 by the constraints $C_1^{x'}, C_2^{x'}, C_{x=x'}$. We get a nice tree decomposition (T', χ', r) of the incidence graph of I' by modifying (T, χ, r) as follows: First we replace in the bags of t'_x and all nodes below t'_x the variable x with x' and constraint C_i with $C_i^{x'}$, $i \in \{1,2\}$. Second we replace the edge $t'_x t_x$ of T with the path $t'_x t_1 t_2 t_3 t_4 t_x$, where $t_1, \dots, t_4$ are new nodes. The bags of these new nodes are defined as in Case 1.

Evidently, (T', χ', r) is a nice tree decomposition of the incidence graph of I'. Since (T, χ, r) is k-special and t'_x is the child of the forget node t_x where variable x is forgotten that occurs in the scopes of more than three constraints, $|\chi(t'_x)| = |\chi'(t'_x)| \leq k+1$. Hence, it follows as in the preceding case that (T', χ', r) is k-special. This completes the proof of the second case.

As long as there are still variables that occur in the scopes of more than three constraints we repeat the above construction until we end up with an instance I^* and a k-special tree decomposition of the incidence graph of I^*. □

In particular, if we apply the splitting procedure of Lemma 4 to the CSP instance I_n as defined in the proof Lemma 3, we obtain a CSP instance of incidence treewidth 3. The corresponding incidence graph for the case $n = 2$ is depicted in Figure 3.

We call a CSP parameter p to be *splitting compatible* if there exists a computable function f such that for every CSP instance I and every instance I^*

obtained from I by splitting we have $p(I^*) \leq f(p(I))$. Note that many natural parameters such as dom, arity, and dep are splitting compatible.

Theorem 2 *Let S be a set of splitting compatible CSP parameters. Then there is an fpt-reduction from* $\mathrm{CSP}(\mathsf{tw}^*, S)$ *to* $\mathrm{CSP}(\mathsf{tw}^d, S)$*.*

PROOF. Let $(I, k, k_1, \ldots, k_r)$ be an instance of $\mathrm{CSP}(\mathsf{tw}^*, p_1, \ldots, p_r)$; i.e., $\mathsf{tw}^*(I) \leq k$ and $p_i(I) \leq k_i$ for $1 \leq i \leq r$. First we compute a nice tree decomposition (T, χ, r) of the incidence graph of I of width at most k. Now we can use Lemma 4 to obtain by splitting the instance I^* and the nice tree decomposition (T', χ', r) of the incidence graph of I^* of width at most $k+1$. From (T', χ', r) we obtain a tree decomposition of the incidence graph of I^* by replacing each variable x in each bag $\chi'(t)$ by the constraints of I^* in which x occurs. Evidently, this produces a tree decomposition (T'', χ'') of the dual graph of I^*. Moreover, since every bag $\chi'(t)$ contains at most $k+1$ variables, the width of (T'', χ'') is at most $3(k+1)$. Let $k' = 3(k+1)$ and $k'_i = f_i(k_i)$, where f_i is the function witnessing that parameter p_i is splitting compatible, $1 \leq i \leq r$. Now $(I^*, k', k'_1, \ldots, k'_r)$ is an instance of $\mathrm{CSP}(\mathsf{tw}^d, p_1, \ldots, p_r)$, hence the result follows. $\square$

The next result, combined with Theorem 2, yields fixed-parameter tractability of $\mathrm{CSP}(\mathsf{tw}^*, \mathsf{dep})$, the first of the three tractability results required for establishing the Classification Theorem.

Theorem 3 $\mathrm{CSP}(\mathsf{tw}^d, \mathsf{dep})$ *is fixed-parameter tractable.*

PROOF. Let I be an instance of $\mathrm{CSP}(\mathsf{tw}^d, \mathsf{dep})$ with $k = \mathsf{tw}^d(I)$ and $p = \mathsf{dep}(I)$. We compute in linear time a tree decomposition (T, χ) of the dual graph of I of width k. Then we compute the join of all constraints in $\chi(t)$ for each node t of T, which yields an equivalent acyclic CSP instance I'. Note that the join of at most $k+1$ constraints can be computed in time $\mathcal{O}(kp^{k+1})$ and there are at most p^{k+1} tuples in each relation of I'. Since k and p are bounded, the above reduction from I to I' is an fpt-reduction. Since I' is acyclic, we can simply apply Yannakakis' algorithm [28]. $\square$

In view of Theorem 2, we immediately obtain the following corollary.

Corollary 3 $\mathrm{CSP}(\mathsf{tw}^*, \mathsf{dep})$ *is fixed-parameter tractable.*

Remark. The construction in the proof of Theorem 2 can be applied to propositional satisfiability in two ways. First, a direct application of the construction to CNF formulas gives an fpt-reduction from SAT(tw^*) to SAT(tw^d) (recall the definitions for SAT(p) as given in Section 1.3). Second, in the context of satisfiability we can also define a splitting operation that splits a clause $(\ell_1 \vee \ell_2 \vee \ell_3 \vee \cdots \vee \ell_r)$ with $r \geq 4$ into two clauses $(\ell_1 \vee \ell_2 \vee x)$ and $(\neg x \vee \ell_3 \cdots \vee \ell_r)$ where x is a new variable. By repeated application of this operation we can transform any CNF formulas into one with at most three literals per clause. A construction dual to the one given in the proof of Theorem 2, splitting clauses instead of variables, yields an fpt-reduction from SAT(tw^*) to SAT(tw).

We continue with further tractability results for the Classification Theorem. What remains to show is fixed-parameter tractability of CSP($\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}$) and CSP($\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}$).

Theorem 4 CSP($\mathsf{tw}^d, \mathsf{dom}, \mathsf{diff}$) *is fixed-parameter tractable.*

PROOF. We proceed in a similar way as for Theorem 3. Let I be an instance of CSP($\mathsf{tw}^d, \mathsf{dom}, \mathsf{diff}$) with $k = \mathsf{tw}^d(I)$, $d = \mathsf{dom}(I)$, and $q = \mathsf{diff}(I)$. We compute in linear time a tree decomposition (T, χ) of the dual graph of I of width k. Then we compute again the join of all constraints in $\chi(t)$ for each node t of T, which yields an equivalent acyclic CSP instance I'. Note that the resulting relation after performing a join-operation on two constraints with relations containing at most p tuples contains at most pd^q tuples. Thus, the join of at most $k+1$ constraints can be computed in time $\mathcal{O}(kp^2d^{qk})$ and there are at most $pd^{q(k+1)}$ tuples in each relation of I'. Since k, d, and q are bounded, the above reduction from I to I' is an fpt-reduction. Since I' is acyclic, we can simply apply Yannakakis' algorithm [28]. □

Again, in view of Theorem 2, we immediately obtain the following corollary.

Corollary 4 CSP($\mathsf{tw}^*, \mathsf{dom}, \mathsf{diff}$) *is fixed-parameter tractable.*

Finally, we are left with our third tractability result.

Theorem 5 CSP($\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}$) *is fixed-parameter tractable.*

PROOF. Let I be an instance of CSP($\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}$) with $d = \mathsf{dom}(I)$, $c = \mathsf{cons}(I)$, and $l = \mathsf{ovl}(I)$. It is easy to see that each constraint has at most $(c-1)l$ variables in its scope that occur also in the scopes of other constraints in I. Thus, for deciding consistency, only these $(c-1)l$ variables are relevant and all others can be projected out. Consequently, we can transform I by an fpt-reduction into an equivalent CSP instance I' with $\mathsf{dom}(I') = d$, $\mathsf{cons}(I') = c$, and $\mathsf{arity}(I') \leq (c-1)l$. Since $\{\mathsf{tw}^d, \mathsf{dep}\}$ dominates $\{\mathsf{dom}, \mathsf{cons}, \mathsf{arity}\}$

by Lemma 2(4) and (12), the tractability of CSP($\mathsf{dom}, \mathsf{cons}, \mathsf{ovl}$) follows by Lemma 1 and Theorem 3. $\square$

3.2 Hardness Results

We first show that CSP($\mathsf{tw}^d, \mathsf{dom}, \mathsf{ovl}$) is W[1]-hard. To this aim, we prove that the problem remains W[1]-hard if we allow only Boolean domain.

Theorem 6 *CSP$_{\mathrm{boole}}$($\mathsf{tw}^d, \mathsf{ovl}$) is* W[1]*-hard.*

PROOF. We give an fpt-reduction from the W[1]-complete problem Independent Set to CSP$_{\mathrm{boole}}$($\mathsf{tw}^d, \mathsf{ovl}$). To this aim, consider a graph $G = (V, E)$ and an integer k. We assume w.l.o.g. that G is nontrivial and connected. We construct a Boolean CSP instance I such that I is consistent if and only if G has an independent set of size k. The instance I is constructed as follows: For every edge $uv \in E$ and $i \in \{1, \ldots, k\}$ we take two variables $x_i[u, v]$ and $x_i[v, u]$. Moreover, for every $i \in \{1, \ldots, k\}$ we construct a "big" constraint C_i of arity $2|E|$, and for every edge $e \in E$ we construct a "small" constraint $C[e]$ of arity $2k$.

In particular, for each $i \in \{1, \ldots, k\}$ we have a big constraint C_i whose scope consists of all variables $x_i[u, v]$ and $x_i[v, u]$ for $uv \in E$. The relation R_i of C_i is defined such that for every satisfying assignment there is exactly one vertex u such that all variables of the form $x_i[u, v]$ are set to 1 and all other variables in the scope of C_i are set to 0 (thus R_i contains exactly $|V|$ tuples). Further, for each edge $uv \in E$ we have a small constraint $C[uv]$ whose scope consists of all the variables $x_i[u, v]$ and $x_i[v, u]$ for $i \in \{1, \ldots, k\}$. The relation $R[uv]$ of $C[uv]$ is defined such that every satisfying assignment sets at most one of the variables in the scope of $C[uv]$ to 1 (thus $R[uv]$ contains exactly $2k+1$ tuples).

Let $S = \{s_1, \ldots, s_k\} \subseteq V$ be an independent set of G of size k. We define an assignment τ setting $\tau(x_i[u, v]) = 1$ if and only if $u = s_i$. It is easy to see that τ satisfies all constraints of I.

Conversely, let τ be an assignment that satisfies I. Because of the big constraints, there is for every $i \in \{1, \ldots, k\}$ exactly one vertex $s_i \in V$ such that τ maps all variables of the form $x_i[s_i, v]$ to 1. We show that $S = \{s_1, \ldots, s_k\}$ is an independent set of size k. First note that $s_i \neq s_j$ for $i \neq j$ since otherwise a small constraint $C[s_i v]$ for some edge $s_i v \in E$ would be invalidated. Hence, $|S| = k$. Second, if there were two adjacent vertices $u, v \in S$, then the small constraint $C[uv]$ would be invalidated. Hence, S is independent.

It remains to show that $\mathsf{tw}^d(I)$ and $\mathsf{ovl}(I)$ are both bounded. Let T be a path with $|E|$ nodes v_e with $e \in E$. For every node v_e we define $\chi(v_e) = \{C[e], C_1, \dots, C_k\}$. It is easy to see that (T, χ) is a tree decomposition of the dual graph of I of width k. Thus, $\mathsf{tw}^d(I) \leq k$. Note that the scopes of any two big constraints (and of any two small constraints) are disjoint. The only non-empty overlap is between a big constraint C_i and a small constraint $C[uv]$ and contains exactly the variables $x_i[u, v]$ and $x_i[v, u]$. Thus, $\mathsf{ovl}(I) = 2$. $\square$

Next we show that CSP($\mathsf{cons}, \mathsf{arity}$) is W[1]-hard. To this aim, we prove that the problem remains W[1]-hard if we allow only binary constraints (i.e., constraints of arity 2). Our reduction was used by Papadimitriou and Yannakakis [22] for showing W[1]-hardness of CSP(vars).

Theorem 7 $\text{CSP}_{\text{bin}}(\mathsf{cons})$ *is* W[1]*-hard.*

PROOF. We give an fpt-reduction from the W[1]-complete problem CLIQUE to $\text{CSP}_{\text{bin}}(\mathsf{cons})$. To this aim, consider a graph $G = (V, E)$ and an integer k. Let $E' = \{(u, v), (v, u) : uv \in E\}$. We construct a binary CSP instance $I = (\{x_1, \dots, x_k\}, V, \mathcal{C})$, where $\mathcal{C}$ contains constraints $((x_i, x_j), E')$ for all $1 \leq i < j \leq k$. Evidently G contains a clique on k vertices if and only if I is consistent. Thus, the theorem follows by observing that $\mathsf{cons}(I) = \binom{k}{2}$. $\square$

Now we show that CSP($\mathsf{dom}, \mathsf{cons}$) is W[1]-hard. Similar to Theorem 6, we prove that the problem remains W[1]-hard if we allow only Boolean domain.

Theorem 8 $\text{CSP}_{\text{boole}}(\mathsf{cons})$ *is* W[1]*-hard.*

PROOF. We give an fpt-reduction from the W[1]-complete problem CLIQUE to $\text{CSP}_{\text{boole}}(\mathsf{cons})$. To this aim, consider a graph $G = (\{v_1, \dots, v_n\}, E)$ and an integer k. We construct a Boolean CSP instance $I = (\{\, x_{i,j} : 1 \leq i \leq k, 1 \leq j \leq n \,\}, \{0, 1\}, \mathcal{C})$ such that I is consistent if and only if there exists a clique of size k in G.

We construct the relation $R \subseteq \{0, 1\}^{2n}$ that encodes the edges of G as follows: For each edge $v_p v_q$ of G, $1 \leq p < q \leq n$, we add to R the $2n$-tuple

$$(t_{p,1}, \dots, t_{p,n}, t_{q,1}, \dots, t_{q,n})$$

where $t_{p,i} = 1$ if and only if $p = i$, and $t_{q,i} = 1$ if and only if $q = i$, $1 \leq i \leq n$. Now let $\mathcal{C}$ be the set of constraints

$$C_{i,j} = ((x_{i,1}, \dots, x_{i,n}, x_{j,1}, \dots, x_{j,n}), R)$$

for $1 \leq i < j \leq k$. It is easy to see that G contains a clique on k vertices if and only if I is consistent. Thus, since $\mathsf{cons}(I) = \binom{k}{2}$, the theorem follows. $\square$

Finally, we are left with the hardness proof of CSP$(\mathsf{dom}, \mathsf{arity}, \mathsf{deg})$. In contrast to the previous hardness results, this problem is actually NP-hard due to a standard reduction from graph 3-colorability. Thus CSP$(\mathsf{dom}, \mathsf{arity}, \mathsf{deg})$ is not fixed-parameter tractable unless P = NP. This even holds if we allow only binary constraints.

Theorem 9 CSP$_{\text{bin}}(\mathsf{dom}, \mathsf{deg})$ *is* NP-*hard.*

PROOF. We give an fpt-reduction from graph 3-colorability to CSP$_{\text{bin}}(\mathsf{dom}, \mathsf{deg})$. To this aim, consider a graph $G = (V, E)$. Note that we can assume w.l.o.g. that each vertex of G has degree at most 4, since graph 3-colorability remains NP-complete under this restriction [13]. We construct a binary CSP instance $I = (V, \{1, 2, 3\}, \mathcal{C})$, where $\mathcal{C}$ contains constraints $((u, v), \{1, 2, 3\}^2 \setminus \{(1, 1), (2, 2), (3, 3)\})$ for all $uv \in E$. Evidently G is 3-colorable if and only if I is consistent. Thus the theorem follows by observing that $\mathsf{dom}(I) = 3$ and $\mathsf{deg}(I) = 4$. $\square$

4 Conclusion

We have presented a general framework for studying the trade-off between generality and performance for parameterized constraint satisfaction. Within our framework we have classified the parameterized complexity of combinations of natural parameters including the treewidth of primal, dual, and incidence graphs, the domain size, the largest size of a constraint relation, and the largest size of a constraint scope. The parameterized complexity of further parameters and their combinations remain open for future research. Furthermore, it would be interesting to extend the hardness results of this paper to completenes results for classes of the weft hierarchy.

Acknowledgment

We thank Moshe Vardi for suggesting us to include the treewidth of dual graphs into our considerations. We also thank an anonymous referee for pointing out the incorrectness of Lemma 3 in the preliminary version [24] of this paper and for suggesting the proof of Theorem 6.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[2] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[3] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1-2):1–45, 1998.

[4] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *Journal of Algorithms*, 21(2):358–402, 1996.

[5] H. Chen and V. Dalmau. Beyond hypertree width: Decomposition methods without decompositions. In *Proc. 11th International Conference on Principles and Practice of Constraint Programming (CP'05)*, vol. 3709 of LNCS, pages 167–181. Springer-Verlag, 2005.

[6] D. Cohen and P. Jeavons. The complexity of constraint languages. In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*, part 1, chapter 8, pages 245–280. Elsevier, 2006.

[7] D. Cohen, P. Jeavons, and M. Gyssens. A unified theory of structural tractability for constraint satisfaction and spread cut decomposition. In *Proc. 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, pages 72–77. Professional Book Center, 2005.

[8] R. Dechter. Tractable structures for constraint satisfaction problems. In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*, part 1, chapter 7, pages 209–244. Elsevier, 2006.

[9] R. Dechter and J. Pearl. Tree clustering for constraint networks. *Artificial Intelligence*, 38(3):353–366, 1989.

[10] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer-Verlag, 1999.

[11] J. Flum and M. Grohe. *Parameterized Complexity Theory.* Springer-Verlag, 2006.

[12] E. C. Freuder. A sufficient condition for backtrack-bounded search. *Journal of the ACM*, 32(4):755–761, 1985.

[13] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976.

[14] G. Gottlob, M. Grohe, N. Musliu, M. Samer, and F. Scarcello. Hypertree decompositions: Structure, algorithms, and applications. In *Proc. 31st International Workshop on Graph-Theoretic Concepts in Computer Science (WG'05)*, vol. 3787 of LNCS, pages 1–15. Springer-Verlag, 2005.

[15] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions: A survey. In *Proc. 26th International Symposium on Mathematical Foundations of Computer Science (MFCS'01)*, vol. 2136 of LNCS, pages 37–57. Springer-Verlag, 2001.

[16] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences*, 64(3):579–627, 2002.

[17] G. Gottlob, F. Scarcello, and M. Sideri. Fixed-parameter complexity in AI and nonmonotonic reasoning. *Artificial Intelligence*, 138(1-2):55–86, 2002.

[18] M. Grohe and D. Marx. Constraint solving via fractional edge covers. In *Proc. 17th ACM-SIAM Symposium on Discrete Algorithms (SODA'06)*, pages 289–298, ACM Press, 2006.

[19] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences* 63(4):512–530, 2001.

[20] P. G. Kolaitis and M. Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Sciences*, 61(2):302–332, 2000.

[21] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.

[22] C. H. Papadimitriou and M. Yannakakis. On the complexity of database queries. *Journal of Computer and System Sciences*, 58(3):407–427, 1999.

[23] B. Reed. Finding approximate separators and computing tree width quickly. In *Proc. 24th ACM Symposium on Theory of Computing (STOC'92)*, pages 221–228. ACM Press, 1992.

[24] M. Samer and S. Szeider. Constraint satisfaction with bounded treewidth revisited. In *Proc. 12th Internationial Conference on Principles and Practice of Constraint Programming (CP'06)*, vol. 4204 of LNCS, pages 499–513. Springer-Verlag, 2006.

[25] M. Samer and S. Szeider. Algorithms for propositional model counting. In *Proc. 14th Internationial Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR'07)*, vol. 4790 of LNCS, pages 484–498. Springer-Verlag, 2007.

[26] M. Samer and S. Szeider. Fixed-parameter tractability. In A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors, *Handbook of Satisfiability*, part 1, chapter 13. IOS Press. To appear.

[27] S. Szeider. On fixed-parameter tractable parameterizations of SAT. In *Proc. 6th International Conference on Theory and Applications of Satisfiability (SAT'03), Selected and Revised Papers*, vol. 2919 of LNCS, pages 188–202. Springer-Verlag, 2004.

[28] M. Yannakakis. Algorithms for acyclic database schemes. In *Proc. 7th International Conference on Very Large Data Bases (VLDB'81)*, pages 82–94. IEEE Computer Society, 1981.