Flow-augmentation III: Complexity dichotomy for Boolean CSPs parameterized by the number of unsatisfied constraints*

Eun Jung Kim[1], Stefan Kratsch[2], Marcin Pilipczuk[3], and Magnus Wahlström[4]

[1]Université Paris-Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, 75016, Paris, France.
[2]Humboldt-Universität zu Berlin, Germany
[3]University of Warsaw, Warsaw, Poland
[4]Royal Holloway, University of London, TW20 0EX, UK

Abstract

We study the parameterized problem of satisfying "almost all" constraints of a given formula $\mathcal{F}$ over a fixed, finite Boolean constraint language Γ, with or without weights. More precisely, for each finite Boolean constraint language Γ, we consider the following two problems. In Min SAT(Γ), the input is a formula $\mathcal{F}$ over Γ and an integer k, and the task is to find an assignment $\alpha\colon V(\mathcal{F}) \to \{0,1\}$ that satisfies all but at most k constraints of $\mathcal{F}$, or determine that no such assignment exists. In Weighted Min SAT(Γ), the input additionally contains a *weight function* $\omega\colon \mathcal{F} \to \mathbb{Z}_+$ and an integer W, and the task is to find an assignment α such that (1) α satisfies all but at most k constraints of $\mathcal{F}$, and (2) the total weight of the violated constraints is at most W. We give a complete dichotomy for the fixed-parameter tractability of these problems: We show that for every Boolean constraint language Γ, either Weighted Min SAT(Γ) is FPT; or Weighted Min SAT(Γ) is W[1]-hard but Min SAT(Γ) is FPT; or Min SAT(Γ) is W[1]-hard. This generalizes recent work of Kim et al. (SODA 2021) which did not consider weighted problems, and only considered languages Γ that cannot express implications $(u \to v)$ (as is used to, e.g., model digraph cut problems). Our result generalizes and subsumes multiple previous results, including the FPT algorithms for Weighted Almost 2-SAT, weighted and unweighted ℓ-Chain SAT, and Coupled Min-Cut, as well as weighted and directed versions of the latter. The main tool used in our algorithms is the recently developed method of *directed flow-augmentation* (Kim et al., STOC 2022).

*This research is a part of a project that have received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme Grant Agreement 714704 (M. Pilipczuk). Eun Jung Kim is supported by the grant from French National Research Agency under JCJC program (ASSK: ANR-18-CE40-0025-01). The full version of the paper can be accessed at `https://arxiv.org/abs/2207.07422`.

1 Introduction

Constraint satisfaction problems (CSPs) are a popular, heavily studied framework that allows a wide range of problems to be expressed and studied in a uniform manner. Informally speaking, a CSP is defined by fixing a domain D and a *constraint language* Γ over D controlling the types of constraints that are allowed in the problem. The problem CSP(Γ) then takes as input a conjunction of such constraints, and the question is whether there is an assignment that satisfies all constraints in the input. Some examples of problems CSP(Γ) for particular constraint languages Γ include 2-SAT, k-Coloring, linear equations over a finite field, and many more. We use SAT(Γ) for the special case of constraints over the Boolean domain $D = \{0, 1\}$.

More precisely, a constraint language over a domain D is a set of finite-arity relations $R \subseteq D^r$ (where r is the arity of R). A *constraint* over a constraint language Γ is formally a pair (X, R), where $R \in \Gamma$ is a relation from the language, say of arity r, and $X = (x_1, \ldots, x_r)$ is an r-tuple of variables called the *scope* of the constraint. We typically write our constraints as $R(X)$ instead of (X, R), or $R(x_1, \ldots, x_r)$ when the individual participating variables x_i need to be highlighted. Let $\alpha\colon X \to D$ be an assignment. Then α *satisfies* the constraint $R(X)$ if $(\alpha(x_1), \ldots, \alpha(x_r)) \in R$, and we say that α *violates* the constraint otherwise. A *formula over* Γ is then a conjunction of constraints over Γ, and the problem CSP(Γ) is to decide, given a formula $\mathcal{F}$ over Γ, whether $\mathcal{F}$ is satisfiable, i.e., if there is an assignment that satisfies all constraints of $\mathcal{F}$. To revisit the examples above, if $D = \{0, 1\}$ is the Boolean domain and Γ contains only relations of arity at most 2 over D, then SAT(Γ) is polynomial-time decidable by reduction to 2-SAT. Similarly, if each relation $R \in \Gamma$ can be defined via linear equations over GF(2), e.g., $R(x_1, \ldots, x_r) \equiv (x_1 + \ldots + x_r = 1 \pmod 2)$, then SAT($\Gamma$) is polynomial-time decidable via Gaussian elimination. Finally, k-Coloring corresponds to CSP(Γ) over a domain $D = \{1, \ldots, k\}$ of cardinality k, and with the constraint language Γ containing only the relation $R \subseteq D^2$ defined as $R(u, v) \equiv (u \neq v)$. Note that these reductions can also easily be turned into equivalences, i.e., there is a specific constraint language Γ such that SAT(Γ) respectively CSP(Γ) is effectively equivalent to 2-SAT, k-Coloring, linear equations over a fixed finite field, and so on.

By capturing such a range of problems in one framework, the CSP framework also allows us to study these problems in a uniform manner. In particular, it allows for the complete characterisation of the complexity of every problem in the framework – so-called *dichotomy theorems*. The most classical is by Schaefer [Sch78], who showed that for every finite Boolean language Γ, either Γ is contained in one of six maximal tractable classes and SAT(Γ) is in P, or else SAT(Γ) is NP-complete. Since then, many other dichotomy theorems have been settled (many of them mentioned later in this introduction). Perhaps chief among them is the general *CSP dichotomy theorem*: For every finite constraint language Γ over a finite domain, the problem CSP(Γ) is either in P or NP-complete. This result was conjectured by Feder and Vardi in the 90's [FV93], and only fully settled a few years ago, independently by Bulatov [Bul17] and Zhuk [Zhu20].

The existence of dichotomy theorems allows us to formally study the question of what makes a problem in a problem category hard – or rather, since hardness appears to be the default state, what makes a problem in a problem category tractable? From a technical perspective, the answer is often phrased algebraically, in terms of algebraic closure properties of the constraint language which describe abstract symmetries of the solution space (see, for example, the collection edited by Krokhin and Zivný [KZ17]). But the answer can also be seen as answering a related question: What algorithmic techniques are required to handle all tractable members of a problem class? In other words, what are the maximal "islands of tractability" in a problem domain, and what algorithmic techniques do they require?

Thus in particular, a dichotomy theorem requires you to both discover all the necessary tools in your algorithmic toolbox, and to hone each of these tools to the maximum generality required by the domain.

As a natural variation on the CSP problem, when a formula $\mathcal{F}$ is not satisfiable, we might look for an assignment under which as few constraints of $\mathcal{F}$ as possible are violated. This defines an *optimization problem* for every language Γ. Formally, for a constraint language Γ, the problem Min CSP(Γ) takes as input a formula $\mathcal{F}$ over Γ and an integer k, and asks if there is an assignment under which at most k constraints of $\mathcal{F}$ are violated. Again, we use Min SAT(Γ) to denote the special case where Γ is over the domain $\{0, 1\}$. Equivalently, we may consider the *constraint deletion* version of Min CSP(Γ) and Min SAT(Γ): Given a formula $\mathcal{F}$ over Γ and integer k, is there a set $Z \subseteq \mathcal{F}$ of at most k constraints such that $\mathcal{F} - Z$ is satisfiable? This version tends to fit better with our algorithms. We refer to such a set Z as *deletion set*.

Let us consider an example. Let $\Gamma = \{(x = 1), (x = 0), (x \to y)\}$. Then Min SAT($\Gamma$) is effectively equivalent to finding a minimum st-cut in a digraph. Indeed, let $\mathcal{F}$ be a formula over Γ and define a digraph G on vertex set $V(G) = V(\mathcal{F}) \cup \{s, t\}$, with an arc (s, v) for every constraint $(v = 1)$ in $\mathcal{F}$, an arc (v, t) for every constraint $(v = 0)$ in $\mathcal{F}$, and an arc (u, v) for every constraint $(u \to v)$ in $\mathcal{F}$. Let $S \subseteq V(G)$ be a vertex set with $s \in S$ and $t \notin S$, and define an assignment $\alpha_S \colon V(\mathcal{F}) \to \{0, 1\}$ by $\alpha(v) = 1$ if and only if $v \in S$. Then the constraints of $\mathcal{F}$ violated by α_S are precisely the edges $\delta_G(S)$ leaving S in G, i.e., an *st*-cut in G. In particular, Min SAT(Γ) is solvable in polynomial time. Naturally, by the same mapping we can also generate an instance $(\mathcal{F}, k)$ of Min SAT(Γ) from a graph G with marked vertices $s, t \in V(G)$, such that $(\mathcal{F}, k)$ is a yes-instance if and only if G has an *st*-cut of at most k edges, justifying the claim that the problems are equivalent.

Unfortunately, for most languages Γ the resulting problem Min SAT(Γ) is NP-hard. Indeed, Khanna et al. [KSTW00] showed that the above example is essentially the only non-trivial tractable case; for every constraint language Γ, either formulas over Γ are always satisfiable for trivial reasons, or Min SAT(Γ) reduces to *st*-Min Cut, or Min SAT(Γ) is APX-hard. (Furthermore, many interesting examples of Min SAT(Γ) do not appear to even allow constant-factor approximations; see discussion below.)

However, Min SAT(Γ) is a natural target for studies in parameterized complexity. Indeed, taking k as a natural parameter, many cases of Min SAT(Γ) have been shown to be FPT when parameterized by k, including the classical problems of Edge Bipartization, corresponding to a language $\Gamma = \{(x \neq y)\}$, and Almost 2-SAT, corresponding to a language Γ containing all 2-clauses. The former is FPT by Reed et al. [RSV04], the latter by Razgon and O'Sullivan [RO09], both classical results in the parameterized complexity literature. It is thus natural to ask for a general characterisation: For which Boolean languages Γ is Min SAT(Γ) FPT parameterized by k?

Indeed, following early FPT work on related CSP optimization problems [Mar05, BM14, KW10, KMW16], the Min SAT(Γ) question was a natural next target. Unfortunately, for a long time this question appeared out of reach, due to some very challenging open problems, specifically Coupled Min-Cut and ℓ-Chain SAT. Coupled Min-Cut is a graph separation problem, never publically posed as an open problem, but long known to be an obstacle to a dichotomy. It was settled to be FPT last year by Kim et al. [KKPW21] as an application of the new *flow augmentation* technique. ℓ-Chain SAT is a digraph cut problem posed by Chitnis et al. in 2013 [CEM13, CEM17], conjectured to be FPT; this conjecture was confirmed only this year by Kim et al. [KKPW22a], as an application of the directed version of flow augmentation. With these obstacles now settled, we find it is time to attack the FPT/W[1] dichotomy question for Min SAT(Γ) directly.

1.1 Our results As mentioned, we consider two variants of Min SAT(Γ), with and without constraint weights. Let Γ be a finite Boolean constraint language. Min SAT(Γ) is the problem defined above: Given input $(\mathcal{F}, k)$, where $\mathcal{F}$ is a conjunction of constraints using relations of Γ, decide if there is a deletion set of cardinality at most k, i.e., a set $Z \subseteq \mathcal{F}$ of at most k constraints such that $\mathcal{F} - Z$ is satisfiable. In the weighted version Weighted Min SAT(Γ), the input is $(\mathcal{F}, \omega, k, W)$, where the formula $\mathcal{F}$ comes equipped with a weight function $\omega \colon \mathcal{F} \to \mathbb{Z}_+$ assigning weights to the constraints of $\mathcal{F}$, and the goal is to find a deletion set of cardinality at most k and weight at most W. Note that this is a fairly general notion of a weighted problem; e.g., it could be that there is an assignment violating strictly fewer than k constraints, but that every such assignment violates constraints to a weight of more than W.

We give a full characterization of Min SAT(Γ) and Weighted Min SAT(Γ) as being either FPT or W[1]-hard when parameterized by the number of violated constraints k. This extends previous partial or approximate FPT-dichotomies of Bonnet et al. [BEM16, BELM18] and Kim et al. [KKPW21].

Theorem 1.1. *Let Γ be a finite Boolean constraint language. Then one of the following applies for the parameterization by the number of unsatisfied constraints.*

1. Weighted Min SAT(Γ) *is FPT.*
2. Min SAT(Γ) *is FPT but* Weighted Min SAT(Γ) *is W[1]-hard.*
3. Min SAT(Γ) *is W[1]-hard.*

Our characterization is combinatorial, and is given in terms of graphs that encode the structure of each constraint. To state it, we first need some terminology. We say that a Boolean relation R is *bijunctive* if it is expressible as a conjunction of 1- and 2-clauses, and R is *IHS-B-* (respectively *IHS-B+*) if it is expressible as a conjunction of negative clauses $(\neg x_1 \vee \ldots \vee \neg x_r)$, positive 1-clauses (x), and implications $(x \to y)$ (respectively positive clauses, negative 1-clauses, and implications). Here, IHS-B is an abbreviation for *implicative hitting set, bounded*. A constraint language Γ is bijunctive, IHS-B+, respectively IHS-B- if every relation $R \in \Gamma$ is bijunctive, IHS-B+, respectively IHS-B-. Finally, Γ is IHS-B if it is either IHS-B+ or IHS-B-. (Note that this is distinct from every relation $R \in \Gamma$ being either IHS-B+ or IHS-B-, since the latter would allow a mixture of, e.g., positive and negative 3-clauses, which defines an NP-hard problem SAT(Γ) [Sch78].)

We will characterise the structure of relations in two ways. Let $R \subseteq \{0,1\}^r$ be a Boolean relation. First, slightly abusing terminology in reusing a term from the literature, we define the *Gaifman graph of* R as an undirected graph G_R on vertex set $[r] = \{1,\ldots,r\}$, where there is an edge $\{i,j\} \in E(G_R)$ if and only if the projection of R onto arguments i and j is non-trivial, i.e., if and only if there are values $b_i, b_j \in \{0,1\}$ such that for every $t \in R$ it is not the case that $t[i] = b_i$ and $t[j] = b_j$. Second, we define the *arrow graph* H_R of R as a directed graph on vertex set $[r]$ where $(i,j) \in E(H_R)$ if $R(x_1,\ldots,x_r)$ implies the constraint $(x_i \to x_j)$ without also implying $(x_i = 0)$ or $(x_j = 1)$. Finally, we say that G_R is $2K_2$*-free* if there is no induced subgraph of G_R isomorphic to $2K_2$. Similarly, the arrow graph H_R is $2K_2$-free if the underlying undirected graph of H_R is $2K_2$-free.

For an illustration, consider the relation $R(x,y,z) \equiv (x=1) \wedge (y=z)$. Let us consider the full set of 2-clauses implied by $R(x,y,z)$, i.e.,

$$R(x,y,z) \models (x \vee y) \wedge (x \vee \neg y) \wedge (x \vee z) \wedge (x \vee \neg z) \wedge (\neg y \vee z) \wedge (y \vee \neg z).$$

where (naturally) clauses such as $(\neg y \vee z)$ could also be written $(y \to z)$. Then G_R is a clique, since every pair of variables is involved in some 2-clause. (Indeed, for the readers familiar with the term Gaifman graph from the literature, if R is bijunctive then the Gaifman graph G_R is precisely the Gaifman graph of the 2-CNF formula consisting of all 2-clauses implied by $R(X)$.) The arrow graph H_R contains the arcs $(2,3)$ and $(3,2)$, due to the last two clauses. On the other hand, despite the 2-clauses $(y \to x)$ and $(z \to x)$ being valid in $R(x,y,z)$, H_R does not contain arcs $(2,1)$ or $(3,1)$ since they are only implied by the assignment $(x = 1)$.

We can now present the FPT results.

THEOREM 1.2. *Let Γ be a finite, bijunctive Boolean constraint language. If for every relation $R \in \Gamma$ the Gaifman graph G_R is $2K_2$-free, then* WEIGHTED MIN SAT(Γ) *is FPT.*

THEOREM 1.3. *Let Γ be a finite, IHS-B Boolean constraint language. If for every $R \in \Gamma$ the arrow graph H_R is $2K_2$-free, then* MIN SAT(Γ) *is FPT.*

We note that Theorem 1.3 encompasses two language classes, corresponding to IHS-B+ or IHS-B-. By symmetry of the problem, the resulting Min SAT problems are essentially equivalent (e.g., by exchanging x and $\neg x$ in all relation definitions); hence it suffices to provide an FPT algorithm for one of the classes. We focus on the IHS-B- case. We also note that for any relation R that is not bijunctive or IHS-B, such as a ternary linear equation over GF(2) or a Horn clause $R(z,y,z) \equiv (x \wedge y \to z)$, any problem MIN SAT($\Gamma$) with $R \in \Gamma$ is either trivially satisfiable or W[1]-hard. Indeed, this follows from previous work [KKPW21, BELM18].

The final dichotomy in Theorem 1.1 now follows from showing that, except for a few simple cases, for any language Γ not covered by Theorem 1.2 the problem WEIGHTED MIN SAT(Γ) is W[1]-hard, and if furthermore Theorem 1.3 does not apply then MIN SAT(Γ) is W[1]-hard.

Let us provide a few illustrative examples.

- First consider the problem MIN SAT(Γ) for the language $\Gamma = \{(x=1),(x=0),R_4\}$, where R_4 is the relation defined by $R_4(a,b,c,d) \equiv (a=b) \wedge (c=d)$. Then the Gaifman graph G_R and the arrow graph H_R both contain $2K_2$'s, hence MIN SAT(Γ) is W[1]-hard. In fact, this problem, together with the directed version $(a \to b) \wedge (c \to d)$, are the fundamental W[1]-hard case of the dichotomy.

  On the other hand, consider $\Gamma' = \{(x=1),(x=0),(x=y)\}$. Then MIN SAT($\Gamma'$) is in P, as observed above. Furthermore, SAT(Γ) and SAT(Γ') are equivalent problems, since any constraint $R_4(a,b,c,d)$ can simply be split into $(a=b)$ and $(c=d)$. For the same reason, MIN SAT(Γ) has a 2-approximation, since breaking up a constraint over R_4 into separate constraints $(a=b)$ and $(c=d)$ at most doubles the number of violated constraints in any assignment. This illustrates the difference in the care that needs to be taken in an FPT/W[1]-dichotomy, compared to approximability results.

- The problem Edge Bipartization corresponds to Min SAT($(x \neq y)$) and Almost 2-SAT to Min SAT($(x \vee y), (x \to y), (\neg x \vee \neg y)$). Since each relation R here is just binary, clearly G_R is $2K_2$-free. Hence Theorem 1.2 generalizes the FPT algorithm for Weighted Almost 2-SAT [KKPW22a].
- Let $(x_1 \to \ldots \to x_\ell)$ be shorthand for the constraint $R(x_1, \ldots, x_\ell) \equiv (x_1 \to x_2) \wedge \ldots \wedge (x_{\ell-1} \to x_\ell)$. Then Min SAT($(x = 1), (x = 0), (x_1 \to \ldots \to x_\ell)$) is precisely the problem ℓ-Chain SAT [CEM17]. For our dichotomy, note that this constraint can also be written $R(x_1, \ldots, x_\ell) \equiv \bigwedge_{1 \leq i < j \leq \ell}(x_i \to x_j)$. Hence for this relation R, both the graphs G_R and H_R are cliques, and ℓ-Chain SAT is contained in both of our tractable classes. This generalizes the FPT algorithm for ℓ-Chain SAT [KKPW22a].
- Now consider a relation $R_{cmc}(a, b, c, d) \equiv (a = b) \wedge (c = d) \wedge (\neg a \vee \neg c)$. Then Min SAT($(x = 1), (x = 0), R_{cmc}$) is known as Coupled Min-Cut, recently shown to be FPT after being a long-standing (implictly) open problem [KKPW21]. Note that the Gaifman graph of R_{cmc} is isomorphic to K_4, hence Theorem 1.2 generalizes this result. The same holds for natural directed variants such as $R'(a, b, c, d) \equiv (a \to b) \wedge (c \to d) \wedge (\neg a \vee \neg c)$. Note that the Gaifman graph $G_{R'}$ is a P_4, i.e., $2K_2$-free. On the other hand, the arrow graphs of both these relations contain $2K_2$'s, hence, e.g., Min SAT($(x = 1), (x = 0), R_{cmc}, (\neg x \vee \neg y \vee \neg z)$) is W[1]-hard.
- For an example in the other direction, consider a relation such as $R(a, b, c, d) \equiv (\neg a \vee \neg b) \wedge (c \to d)$. Then the Gaifman graph G_R is a $2K_2$, but the arrow graph H_R contains just one edge, showing that Min SAT($(x = 1), (x = 0), R$) is FPT but Weighted Min SAT($(x = 1), (x = 0), R$) is W[1]-hard. Similarly, adding a constraint such as $(x \neq y)$ to the language yields a W[1]-hard problem Min SAT(Γ). Intuitively, this is because having access to variable negation allows us to transform R to the "double implication" constraint $R'(a, b, c, d) \equiv R(a, \neg b, c, d)$ from the W[1]-hard case mentioned above. Indeed, a lot of the work of the hardness results in this paper is to leverage expressive power of Min SAT(Γ) and Weighted Min SAT(Γ) to "simulate" negations, in specific and restricted ways, when $(x \neq y)$ is not available in the language.

1.2 Previous dichotomies and related work Many variations of SAT and CSP with respect to decision and optimization problems have been considered, and many of them are relevant to the current work.

Khanna et al. [KSTW00] considered four optimization variants of SAT(Γ) on the Boolean domain, analyzed with respect to approximation properties. They considered Min Ones(Γ), where the goal is to find a satisfying assignment with as few variables set to 1 as possible; Max Ones(Γ), where the goal is to find a satisfying assignment with as many variables set to 1 as possible; Min SAT(Γ), where the goal is to find an assignment with as few violated constraints as possible; and Max SAT(Γ), where the goal is to find an assignment with as many satisfied constraints as possible. They characterized the P-vs-NP boundary and the approximability properties of all problems in all four variants.

Note that although, e.g., Min SAT(Γ) and Max SAT(Γ) are equivalent with respect to the optimal assignments, from a perspective of approximation they are very different. Indeed, for any finite language Γ, you can (on expectation) produce a constant-factor approximation algorithm for Max SAT(Γ) simply by taking an assignment chosen uniformly at random. For the same reason, Max SAT(Γ) parameterized by the number of satisfied constraints k is trivially FPT, and in fact has a linear kernel for every finite language Γ [KMW16]. On the other hand, Min SAT(Γ) is a far more challenging problem from an approximation and fixed-parameter tractability perspective. In fact, combining the characterisation of Min SAT(Γ) approximability classes of Khanna et al. [KSTW00] with results assuming the famous *unique games conjecture* (UGC; or even the weaker *Boolean unique games conjecture* [EM22]), we find that if the UGC is true, then the only cases of Min SAT(Γ) that admit a constant-factor approximation are when Γ is IHS-B.

The first parameterized CSP dichotomy we are aware of is due to Marx [Mar05], who considered the problem Exact Ones(Γ): Given a formula $\mathcal{F}$ over Γ, is there a satisfying assignment that sets *precisely* k variables to 1? Marx gives a full dichotomy for Exact Ones(Γ) as being FPT or W[1]-hard parameterized by k, later extended to general non-Boolean languages with Bulatov [BM14]. Marx also notes that Min Ones(Γ) is FPT by a simple branching procedure for every finite language Γ [Mar05]. However, the existence of so-called *polynomial kernels* for Min Ones(Γ) problems is a non-trivial question; a characterization for this was given by Kratsch and Wahlström [KW10], and follow-up work mopped up the questions of FPT algorithms and polynomial kernels parameterized by k for all three variants Min/Max/Exact Ones(Γ) [KMW16]

Polynomial kernels for Min SAT(Γ) have been considered; most notably, there are polynomial kernels for Edge Bipartization and Almost 2-SAT [KW20]. We are not aware of any significant obstacles towards a kernelizability dichotomy of Min SAT(Γ); however, a comparable result for more general Min CSP(Γ) appears far out of reach already for a domain of size 3.

Other, direct predecessor results of the current dichotomy include the characterization of fixed-parameter constant-factor approximation algorithms for Min SAT(Γ) of Bonnet et al. [BEM16] and a partial FPT/W[1] dichotomy of Kim et al. [KKPW21]. In particular, the latter paper provided an FPT/W[1]-dichotomy for all cases of Min SAT(Γ) where Γ is unable to express the "directed edge" constraint $(u \to v)$. Finally, we recall that the problem ℓ-Chain SAT was first published by Chitnis et al. [CEM17], who related its status to a conjectured complexity dichotomy for the Vertex Deletion List H-Coloring problem. This conjecture was subsequently confirmed with the FPT-algorithm for ℓ-Chain SAT [KKPW22a].

A much more ambitious optimization variant of CSPs are *Valued CSP*, VCSP. In this setting, instead of a constraint language one fixes a finite set S of *cost functions*, and considers the problem VCSP(S), of minimising the value of a sum of cost functions from S. The cost functions can be either finite-valued or *general*, taking values from $\mathbb{Q} \cup \{\infty\}$ to simulate *crisp*, unbreakable constraints. Both Min Ones(Γ) and Min CSP(Γ) (and, indeed, Vertex Deletion List H-Coloring) are special cases of VCSPs, as are many other problems. The classical (P-vs-NP) complexity of VCSPs admits a remarkably clean dichotomy: There is a canonical LP-relaxation, the *basic LP*, such that for any finite-valued S, the problem VCSP(S) is in P if and only if the basic LP is integral [TZ16]. A similar, complete characterization for general-valued VCSP(S) problems is also known [KKR17].

1.3 Technical overview The technical work of the paper is divided into three parts. Theorem 1.2, i.e., the FPT algorithm for bijunctive languages Γ where every relation $R \in \Gamma$ has a $2K_2$-free Gaifman graph; Theorem 1.3, i.e., the FPT algorithm for IHS-B languages Γ where every relation $R \in \Gamma$ has a $2K_2$-free arrow graph; and the completion of the dichotomy, where we prove that all other cases are trivial or hard. We begin with the algorithmic results.

Graph problem. In both our algorithmic results, we cast the problem at hand as a graph separation problem, which we call Generalized Bundled Cut. An instance consists of

- a directed multigraph G with distinguished vertices $s, t \in V(G)$;
- a multiset $\mathcal{C}$ of subsets of $V(G)$, called *clauses*;
- a family $\mathcal{B}$ of pairwise disjoint subsets of $E(G) \cup \mathcal{C}$, called *bundles*, such that one bundle does not contain two copies of the same arc or clause;
- a parameter k;
- in the weighted variant, additionally a weight function $\omega : \mathcal{B} \to \mathbb{Z}_+$ and a weight budget $W \in \mathbb{Z}_+$.

We seek a set $Z \subseteq E(G)$ that is an st-cut (i.e., cuts all paths from s to t). An edge e is *violated* by Z if $e \in Z$ and a clause $C \in \mathcal{C}$ is *violated* by Z if all elements of C are reachable from s in $G - Z$ (i.e., a clause is a request to separate at least one of the elements from s). A bundle is *violated* if any of its elements is violated. An edge, a clause, or a bundle is *satisfied* if it is not violated. An edge or a clause is *soft* if it is in a bundle and *crisp* otherwise. We ask for an st-cut Z that satisfies all crisp edges and clauses, and violates at most k bundles (and whose total weight is at most W in the weighted variant). An instance is *b-bounded* if every clause is of size at most b and, for every $B \in \mathcal{B}$, the set of vertices involved in the elements of B is of size at most b.

Generalized Bundled Cut, in full generality, can be easily seen to be W[1]-hard when parameterized by k, even with $\mathcal{O}(1)$-bounded instances; see Marx and Razgon [MR09] and the problem Paired Minimum s,t-Cut(ℓ) defined in the full version of the current paper [KKPW22b]. In both tractable cases, the obtained instances are b-bounded for some b depending on the language and have some additional properties that allow for fixed-parameter algorithms when parameterized by $k + b$.

In the reduction, the source vertex s should be interpreted as "true" and the sink vertex t as "false"; other vertices are in 1-1 correspondence with the variables of the input instance. Furthermore, arcs are implications that correspond to parts of the constraints of the input instance. The sought st-cut Z corresponds to implications violated by the sought assignment in the CSP instance; a vertex is assigned 1 in the sought solution if and only if it is reachable from s in $G - Z$.

Thus, in terms of constraints, arcs (u, v) correspond to implications $(u \to v)$ in the input formula, and clauses $\{v_1, \ldots, v_r\}$ correspond to negative clauses $(\neg v_1 \vee \ldots \vee \neg v_r)$ in the input formula. Thereby, each bundle naturally encodes an IHS-B- constraint. Capturing bijunctive constraints requires a little bit more work, since they can also involve positive 2-clauses $(u \vee v)$, but this can be reduced to the IHS-B- case with clauses of arity 2 via standard methods (e.g., iterative compression followed by variable renaming [KW20]). Thus we proceed assuming that the Min SAT(Γ) instance has been reduced to an instance of Generalized Bundled Cut (with the appropriate additional properties eluded to above).

Flow-augmentation. Let us first recall the basics of flow-augmentation of Kim et al. [KKPW22a]. Recall that in Generalized Bundled Cut, we are interested in a deletion set $Z \subseteq E(G)$ that separates t and possibly some more vertices of G from s. Formally, Z is a *star st-cut* if it is an st-cut and additionally for every $(u, v) \in Z$, u is reachable from s in $G - Z$ but v is not. That is, Z cuts all paths from s to t and every edge of Z is essential to separate some vertex of G from s. For a star st-cut Z, its *core*, denoted $\mathrm{core}_G(Z)$, is the set of those edges $(u, v) \in Z$ such that t is reachable from v in $G - Z$. That is, $\mathrm{core}_G(Z)$ is the unique inclusion-wise minimal subset of Z that is an st-cut. A simple but crucial observation is that in Generalized Bundled Cut any inclusion-wise minimal solution Z is a star st-cut.

Considered restrictions of Generalized Bundled Cut turn out to be significantly simpler if the sought star st-cut Z satisfies the following additional property: $\mathrm{core}_G(Z)$ is actually an st-cut of minimum possible cardinality. This is exactly the property that the flow-augmentation technique provides.

Theorem 1.4. (directed flow-augmentation [KKPW22a]) *There exists a polynomial-time algorithm that, given a directed graph G, vertices $s, t \in V(G)$, and an integer k, returns a set $A \subseteq V(G) \times V(G)$ and a maximum flow $\widehat{\mathcal{P}}$ from s to t in $G + A$ such that for every star st-cut Z in G of size at most k, with probability $2^{-\mathcal{O}(k^4 \log k)}$, the sets of vertices reachable from s in $G - Z$ and $(G + A) - Z$ are equal (in particular, Z remains a star st-cut in $G + A$), $\mathrm{core}_{G+A}(Z)$ is an st-cut of minimum possible cardinality, and every flow path of $\widehat{\mathcal{P}}$ contains exactly one edge of Z.*

We call such a flow $\widehat{\mathcal{P}}$ a *witnessing flow*. Note that each path of $\widehat{\mathcal{P}}$ is obliged to contain exactly one edge of $\mathrm{core}_{G+A}(Z)$ as $\mathrm{core}_{G+A}(Z)$ is an st-cut of minimum cardinality and $\widehat{\mathcal{P}}$ is a maximum flow. However, we additionally guarantee that $\widehat{\mathcal{P}}$ does not use any edge of $Z \setminus \mathrm{core}_{G+A}(Z)$.

Bijunctive case. The most complex result is for tractable bijunctive languages. This can be seen as a maximally general generalization of the algorithm for Weighted Almost 2-SAT of previous work [KKPW22a]. Here, the obtained instances of Generalized Bundled Cut satisfy the following requirements: They are b-bounded, where b is the maximum arity in Γ, all clauses are of size 2, and for every bundle $B \in \mathcal{B}$ the following graph G_B is $2K_2$-free: $V(G_B)$ consists of all vertices involved in an element of B except for s and t, and $uv \in E(G_B)$ if there is a clause $\{u, v\} \in B$, an arc $(u, v) \in B$, or an arc $(v, u) \in B$. The weighted version of the Generalized Bundled Cut with these extra requirements is called Generalized Digraph Pair Cut, and settling the fixed-parameter tractability of the latter problem is handled in Section 3 of the full version of the paper [KKPW22b].

We give an overview of the algorithm for Generalized Digraph Pair Cut. Suppose that the given instance $\mathcal{I} = (G, s, t, \mathcal{C}, \mathcal{B}, \omega, k, W)$ is a Yes-instance with a solution Z of weight at most W and let $\kappa := |Z|$, i.e. the number of edges violated by Z, x and let κ_c be the number of clauses violated by Z. As previously observed, we can assume that Z is a star st-cut and via Theorem 1.4, we can further assume (with good enough probability) that $\mathrm{core}_G(Z)$ is an st-cut with minimum cardinality and additionally an st-maxflow $\widehat{\mathcal{P}}$ (with $\lambda_G(s, t)$ flow paths) that is a witnessing flow for Z is given. The ethos of the entire algorithm is that we use the witnessing flow $\widehat{\mathcal{P}}$ at hand as a guide in search for Z and narrow down the search space. The algorithm is randomized and at each step, the success probability is at least $2^{-\mathrm{poly}(b,k)}$. Below, we assume that all the guesses up to that point are successful.

In the first stage of the algorithm (see Section 3.1 of [KKPW22b]), the goal is to obtain an instance $\mathcal{I}'$ of Generalized Digraph Pair Cut so that there is an optimal solution Z' to $\mathcal{I}'$ that is an st-cut of G' of minimum cardinality. As $\kappa + \kappa_c \leq 2kb^2$, each of these integers can be correctly guessed with high probability. We may assume that $\kappa > \lambda_G(s, t) = |\mathrm{core}_G(Z)|$; if not, we may either output a trivial No-instance or proceed with the current instance $\mathcal{I}$ as the desired instance. When $Z \setminus \mathrm{core}_G(Z) \neq \emptyset$, this is because there is a clause $p \in \mathcal{C}$ that is violated by $\mathrm{core}_G(Z)$ and some of the extra edges in $Z \setminus \mathrm{core}_G(Z)$ are used to separate an endpoint (called an *active vertex*) of p from s to satisfy p. Even though we cannot sample an active vertex with high enough probability, i.e. $1/f(k,b)$ for some f, we are able to sample a *monotone* sequence of vertices $u_1,\dots,u_\ell$ with high probability in the following sense: There exists a unique active vertex u_a among the sequence, and all vertices before u_a are reachable from s in $G-Z$ and all others are unreachable from s in $G-Z$. Once such a sequence is sampled, the st-path visiting (only) $u_1,\dots,u_\ell$ in order is added as soft arcs, each forming a singleton bundle of weight $W+1$, and we increase the budgets k and W to $k+1$ and $2W+1$ respectively. Let $\mathcal{I}'$ be the new instance. Then $Z' := Z \cup \{(u_{a-1},u_a)\}$ (where $u_0 = s$) is a solution to $\mathcal{I}'$ violating at most $k+1$ bundles with weight at most $2W+1$. The key improvement is here that u_a is connected to t with a directed path in $G-Z'$, implying that at least one of the extra edges in $Z \setminus \mathrm{core}_G(Z)$ used to cut u_a from s (in addition to (u_{a-1},u_a)) is now incorporated into $\mathrm{core}_{G'}(Z')$, thus the value $|Z \setminus \mathrm{core}_G(Z)|$ strictly drops in the new instance. After performing this procedure of "sampling a (monotone) sequence then adding it as an st-path" a bounded number of times, we get a Yes-instance that has an st-mincut as an optimal solution with high enough probability.

We proceed with an instance $\mathcal{I}$ and an st-maxflow $\widehat{\mathcal{P}}$ obtained from the previous stage, i.e., for which there is an optimal solution Z that is an st-mincut whenever $\mathcal{I}$ is a Yes-instance. In the second stage of the algorithm (presented in Section 3.2 of [KKPW22b]), we branch into $f(k)$ instances each of which is either outright No-instance or *ultimately bipartite* in the following sense. Suppose that an instance $\mathcal{I}$ admits a vertex bipartition $V(G) \setminus \{s,t\} = V_0 \uplus V_1$ such that (i) the vertices of any flow path P_i are fully contained in $V_\alpha \cup \{s,t\}$ for some $\alpha \in \{0,1\}$, (ii) for every edge $(u,v) \in E(G)$, we have $u,v \in V_\alpha \cup \{s,t\}$ for some $\alpha \in \{0,1\}$, and (iii) there is no clause $\{u,v\} \in \mathcal{C}$ with $\{u,v\} \subseteq V_\alpha \cup \{s,t\}$ for some $\alpha \in \{0,1\}$. If the instance at hand is ultimately bipartite, we can eliminate the clauses altogether: Reverse the orientations of all edges in $G_1 := G[V_1 \cup \{s,t\}]$ and identify t (s resp.) of the reversed graph with s (t resp.) of $G_0 := G[V_0 \cup \{s,t\}]$, and for any clause $\{u,v\}$ with $u \in V_0$ and $v \in V_1$, replace the clause $\{u,v\}$ by the arc (u,v). Note that the reversal of G_1 converts the property (in $G-Z$) of "v being reachable / unreachable from s" to "v reaching / not reaching t" for all $v \in V(G_1)$. Hence, converting a clause $\{u,v\}$ to an arc (u,v) preserves the same set of violated vertex pairs, except that the violated clauses now become violated edges.

After the aforementioned transformation into an instance without clauses (see Section 3.2.5 of [KKPW22b]), the bundles maintain $2K_2$-freeness. Now, $2K_2$-freeness is used to argue that (after some preprocessing and color-coding steps) the bundles are *ordered* along the flow paths, which allows us to transform (see Section 3.3 of [KKPW22b]) the instance to an instance of the Weighted Minimum st-Cut problem: Given an edge-weighted directed graph G with s and t, find a minimum weight st-cut of cardinality λ, where λ is the cardinality of a maximum st-flow. This problem is polynomial-time solvable. In Section 3.2.1 up to Section 3.2.4 of the full version [KKPW22b], we present the branching strategies and preprocessing steps to reach the ultimately bipartite instances.

IHS-B case. This algorithm can be seen as a mix of the Chain SAT algorithm of [KKPW22a] and the Digraph Pair Cut algorithm of [KW20]. Here, the obtained instances of Generalized Bundled Cut are unweighted and satisfy the following requirements: They are b-bounded, where b is the maximum arity in Γ, and for every bundle $B \in \mathcal{B}$ the following graph G'_B is $2K_2$-free: $V(G'_B)$ consists of all vertices involved in an element of B except for s and t, and $uv \in E(G'_B)$ if there is an arc $(u,v) \in B$ or an arc $(v,u) \in B$. That is, clauses can be larger than 2 (but of size at most b), but in a bundle all arcs not incident with s nor t form a $2K_2$-free graph. By applying flow-augmentation, we can assume that for the sought solution Z it holds that $\mathrm{core}_G(Z)$ is an st-cut of minimum cardinality and we have access to a witnessing flow $\widehat{\mathcal{P}}$. Every path $P \in \widehat{\mathcal{P}}$ contains a unique edge $e_P \in E(P) \cap Z$.

We perform the following color-coding step. For every bundle B and every $P \in \widehat{\mathcal{P}}$ we randomly guess a value $e(B,P) \in \{\perp\} \cup (E(P) \cap B)$. We aim for the following: For every B violated by Z, and every $P \in \widehat{\mathcal{P}}$, we want $e(B,P)$ to be the unique edge of $B \cap E(P) \cap Z$, or $\perp$ if there is no such edge. We also branch into how the edges e_P are partitioned into bundles. That is, for every two distinct $P,P' \in \widehat{\mathcal{P}}$, we guess if e_P and $e_{P'}$ are from the same bundle. Note that this guess determines the number of bundles that contain an edge of $\mathrm{core}_G(Z)$; we reject a guess if this number is larger than k.

Assume that we guessed that $e_P, e_{P'} \in B$ for some $P,P' \in \widehat{\mathcal{P}}$ and a bundle B violated by Z. Then, as G'_B is $2K_2$-free, either e_P or $e_{P'}$ is incident with s or t, or they have a common endpoint, or there is an arc of B from an endpoint of e_P to an endpoint of $e_{P'}$, or there is an arc of B from an endpoint of $e_{P'}$ to an endpoint of e_P. We guess which cases apply. If e_P or $e_{P'}$ is incident with s or t, there is only a constant number of candidates for B, we guess it, delete it, decrease k by one, and restart the algorithm. All later cases are very similar to each other; let us describe here the case that B contains an arc f from an endpoint of e_P to an endpoint of $e_{P'}$.

Let B_1 and B_2 be two bundles that are candidates for bundle B; that is, for $i = 1, 2$ we have $e_i := e(B_i, P) \neq \perp$, $e'_i := e(B_i, P') \neq \perp$, and B_i contains an arc f_i that is a candidate for f: has its tail in e_i and its head in e'_i. Assume that e_1 is before e_2 on P but e'_1 is after e'_2 on P'. The crucial observation (and present also in the algorithm for Chain SAT of [KKPW22a]) is that it cannot hold that $B = B_2$, as if we cut e_2 and e'_2, the edge $f_1 \in B_1$ will provide a shortcut from a vertex on P before the cut to a vertex on P' after the cut. Thus, we may not consider B_2 as a candidate for the bundle B violated by Z. Furthermore, the remaining candidates for the bundle B are linearly ordered along all flow paths P where $B \cap Z \cap E(P) \neq \emptyset$.

This allows the following filtering step: For every bundle B, we expect that $\{P \mid P \in \widehat{\mathcal{P}}, e(B, P) \neq \perp\}$ is consistent with the guessing and there is no other bundle B' with the same set $\{P \mid e(B, P) \neq \perp\}$ that proves that B is not violated by Z as in the previous paragraph. We also expect that $\{e(B, P) \mid P \in \widehat{\mathcal{P}}\}$ extends to a minimum st-cut. If B does not satisfy these conditions, we delete it from $\mathcal{B}$ (making all its edges and clauses crisp).

Now, a simple submodularity argument shows that the first (closest to s) minimum st-cut Z_0 (where only edges $e(B, P)$ for some $B \in \mathcal{B}$ and $P \in \widehat{\mathcal{P}}$ are deletable) has the correct structure: its edges are partitioned among bundles as guessed. If Z_0 is a solution, we return Yes; note that this is the step where we crucially rely on the instance being unweighted. Otherwise, there is a clause $C \in \mathcal{C}$ violated by Z_0. It is either indeed violated by the sought solution Z or there is $v \in C$ that is not reachable from s in $G - Z$. We guess which option happens: In the first case, we delete the bundle containing C, decrease k by one, and restart the algorithm. In the second case, we guess v and add a crisp arc (v, t), increasing the size of a minimum st-cut, and restart the algorithm. This concludes the overview of the algorithm; note that the last branching step is an analog of the core branching step of the Digraph Pair Cut algorithm of [KW20].

Dichotomy completion. Finally, in order to complete the dichotomy we need to show that the above two algorithms cover all interesting cases. This builds on the previous, partial characterization of Kim et al. [KKPW21]. Section 5 of the full version [KKPW22b] contains full proofs for completeness, but for the purposes of this overview we can start from a result from the extended preprint version of a paper of Bonnet et al. [BELM18]. Specifically, they show that for any language Γ that is not IHS-B or bijunctive, Min SAT(Γ) does not even admit a constant-factor approximation in FPT time, parameterized by k. Clearly, there in particular cannot exist exact FPT-algorithms for such languages, hence we may assume that Γ is bijunctive or IHS-B. By a structural observation, we show that either Γ implements positive and negative assignments, i.e., constraints $(x = 1)$ and $(x = 0)$, or Weighted Min SAT(Γ) is trivial in the sense that setting all variables to 1 (respectively to 0) is always optimal. Hence we assume that Γ implements assignments. Furthermore, recall that our basic W[1]-hard constraints are $R_4(a, b, c, d) \equiv (a = b) \wedge (c = d)$ or its variants with one or both equalities replaced by implications.

First assume that Γ is bijunctive and not IHS-B. In particular, every relation $R \in \Gamma$ can be expressed as a conjunction of 1- and 2-clauses, but it does not suffice to use only conjunctions over $\{(x \vee y), (x \to y), (\neg x)\}$ or over $\{(\neg x \vee \neg y), (x \to y), (x)\}$. By standard methods (e.g., the structure of *Post's lattice* [Pos41]), we may then effectively assume that Γ contains the relation $(x \neq y)$. Furthermore, we assume that there is a relation $R \in \Gamma$ such that the Gaifman graph G_R contains a $2K_2$. In fact, assume for simplicity that R is 4-ary and that G_R has edges $\{1, 2\}$ and $\{3, 4\}$. Then R must be a "product" $R(a, b, c, d) \equiv R_1(a, b) \wedge R_2(c, d)$, where furthermore neither R_1 nor R_2 implies an assignment, as such a case would imply further edges of the Gaifman graph. It is now easy to check that each of R_1 and R_2 is either $R_i(x, y) \equiv (\sim x = \sim y)$ or $R_i(x, y) \equiv (\sim x \to \sim y)$, where $\sim v$ represents either v or $\neg v$. It is now not difficult to use R in combination with $\neq$-constraints to implement a hard relation such as R_4, implying W[1]-hardness.

Next, assume that Γ is IHS-B, say ISH-B-, but not bijunctive. Then, again via Post's lattice, we have access to negative 3-clauses $(\neg x \vee \neg y \vee \neg z)$. We first show that either Γ implements equality constraints $(x = y)$, or Weighted Min SAT(Γ) has a trivial FPT branching algorithm. We then need to show that Weighted Min SAT($(x = 1), (x = 0), (x = y), (\neg x \vee \neg y \vee \neg z)$) is W[1]-hard, and that Min SAT(Γ) is W[1]-hard if Theorem 1.3 does not apply. To describe these W[1]-hardness proofs, we need a more careful review of the hardness reduction for Min SAT($(x = 1), (x = 0), R_4$). As is hopefully clear from our discussions, this problem corresponds to finding an st-cut in an auxiliary multigraph G, where the edges of G come in pairs and the cut may use edges of at most k different pairs. We show W[1]-hardness of a further restricted problem, Paired Minimum st-cut, where furthermore the edges of the graph G are partitioned into $2k$ st-paths, i.e., the st-flow in G is precisely $2k$

and any solution needs to cut every path in precisely one edge.

The remaining hardness now all use the same basic idea. Say that a pair of st-paths $P = (s = x_1 = \ldots = x_n = t)$ and $P' = (s = x'_n = \ldots = x'_1 = t)$ in a formula $\mathcal{F}$ over Γ are *complementary* if any min-cost solution to $\mathcal{F}$ cuts between x_i and x_j if and only if it cuts between x'_j and x'_i. In other words, for a min-cost assignment α, we have $\alpha(x_i) \neq \alpha(x'_i)$ for every $i \in [n]$. This way, for the purposes of a hardness reduction from Paired Minimum st-Cut only, we can act as if we have access to $\neq$-constraints by implementing every path in the input instance as a pair of complementary paths over two sets of variables x_v, x'_v in the output formula. Indeed, consider a pair $\{\{u, v\}, \{p, q\}\}$ of edges in the input instance, placed on two distinct st-paths. To force that the pair is cut simultaneously, we wish to use crisp clauses such as $(u \wedge \neg v \to p)$ and $(u \wedge \neg v \to \neg q)$, enforcing that if $\{u, v\}$ is cut, i.e., $\alpha(u) = 1$ and $\alpha(v) = 0$ for the corresponding assignment α, then $\alpha(p) = 1$ and $\alpha(q) = 0$ as well. This is now equivalent to the negative 3-clauses $(\neg u \vee \neg v' \vee \neg p)$ and $(\neg u \vee \neg v' \vee \neg q')$.

We can implement such complementary path pairs in two ways, either with equality, negative 2-clauses, and carefully chosen constraint weights, for hardness of Weighted Min SAT(Γ), or with a relation R such that the arrow graph H_R contains a $2K_2$, for the unweighted case. Here, although the truth is a bit more complex, we can think of such a relation R as representing either R_4 or a constraint such as the coupled min-cut constraint $R(a, b, c, d) \equiv (a \to b) \wedge (c \to d) \wedge (\neg a \vee \neg c)$. Note that the construction of complementary path pairs using such a constraint is straight forward.

The final case, when Γ is both bijunctive and IHS-B-, works via a similar case distinction, but somewhat more complex since we need to consider the interaction of the two theorems Theorem 1.2 and Theorem 1.3. The full characterization of Min SAT(Γ) and Weighted Min SAT(Γ) as FPT or W[1]-hard, including explicit lists of the FPT cases, is found in Lemma 5.22 of the full version [KKPW22b].

References

[BELM18] Édouard Bonnet, László Egri, Bingkai Lin, and Dániel Marx. Fixed-parameter approximability of Boolean MinCSPs. *CoRR*, abs/1601.04935v2, 2018.

[BEM16] Édouard Bonnet, László Egri, and Dániel Marx. Fixed-parameter approximability of Boolean MinCSPs. In *ESA*, volume 57 of *LIPIcs*, pages 18:1–18:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[BM14] Andrei A. Bulatov and Dániel Marx. Constraint satisfaction parameterized by solution size. *SIAM J. Comput.*, 43(2):573–616, 2014.

[Bul17] Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *FOCS*, pages 319–330. IEEE Computer Society, 2017.

[CEM13] Rajesh Hemant Chitnis, László Egri, and Dániel Marx. List H-coloring a graph by removing few vertices. In Hans L. Bodlaender and Giuseppe F. Italiano, editors, *Algorithms - ESA 2013 - 21st Annual European Symposium, Sophia Antipolis, France, September 2-4, 2013. Proceedings*, volume 8125 of *Lecture Notes in Computer Science*, pages 313–324. Springer, 2013.

[CEM17] Rajesh Chitnis, László Egri, and Dániel Marx. List H-coloring a graph by removing few vertices. *Algorithmica*, 78(1):110–146, 2017.

[EM22] Ronen Eldan and Dana Moshkovitz. Reduction from non-unique games to boolean unique games. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 64:1–64:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[FV93] Tomás Feder and Moshe Y. Vardi. Monotone monadic SNP and constraint satisfaction. In *STOC*, pages 612–622. ACM, 1993.

[KKPW21] Eun Jung Kim, Stefan Kratsch, Marcin Pilipczuk, and Magnus Wahlström. Solving hard cut problems via flow-augmentation. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 149–168. SIAM, 2021.

[KKPW22a] Eun Jung Kim, Stefan Kratsch, Marcin Pilipczuk, and Magnus Wahlström. Directed flow-augmentation. In *STOC*, pages 938–947. ACM, 2022.

[KKPW22b] Eun Jung Kim, Stefan Kratsch, Marcin Pilipczuk, and Magnus Wahlström. Flow-augmentation III: complexity dichotomy for boolean CSPs parameterized by the number of unsatisfied constraints. *CoRR*, abs/2207.07422, 2022.

[KKR17] Vladimir Kolmogorov, Andrei A. Krokhin, and Michal Rolínek. The complexity of general-valued CSPs. *SIAM J. Comput.*, 46(3):1087–1110, 2017.

[KMW16] Stefan Kratsch, Dániel Marx, and Magnus Wahlström. Parameterized complexity and kernelizability of Max Ones and Exact Ones problems. *ACM Trans. Comput. Theory*, 8(1):1:1–1:28, 2016.

[KSTW00] Sanjeev Khanna, Madhu Sudan, Luca Trevisan, and David P. Williamson. The approximability of constraint satisfaction problems. *SIAM J. Comput.*, 30(6):1863–1920, 2000.

[KW10] Stefan Kratsch and Magnus Wahlström. Preprocessing of Min Ones problems: A dichotomy. In *ICALP (1)*, pages 653–665, 2010.

[KW20] Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *J. ACM*, 67(3):16:1–16:50, 2020.

[KZ17] Andrei A. Krokhin and Stanislav Zivný, editors. *The Constraint Satisfaction Problem: Complexity and Approximability*, volume 7 of *Dagstuhl Follow-Ups*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[Mar05] Dániel Marx. Parameterized complexity of constraint satisfaction problems. *Computational Complexity*, 14(2):153–183, 2005.

[MR09] Dániel Marx and Igor Razgon. Constant ratio fixed-parameter approximation of the edge multicut problem. *Inf. Process. Lett.*, 109(20):1161–1166, 2009.

[Pos41] Emil L. Post. *The Two-Valued Iterative Systems of Mathematical Logic. (AM-5)*. Princeton University Press, 1941.

[RO09] Igor Razgon and Barry O'Sullivan. Almost 2-SAT is fixed-parameter tractable. *J. Comput. Syst. Sci.*, 75(8):435–450, 2009.

[RSV04] Bruce A. Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Oper. Res. Lett.*, 32(4):299–301, 2004.

[Sch78] Thomas J. Schaefer. The complexity of satisfiability problems. In *STOC*, pages 216–226. ACM, 1978.

[TZ16] Johan Thapper and Stanislav Zivny. The complexity of finite-valued CSPs. *J. ACM*, 63(4):37:1–37:33, 2016.

[Zhu20] Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *J. ACM*, 67(5):30:1–30:78, 2020.