Parameterized Complexity of Equality MinCSP

George Osipov
Department of Computer and Information Science, Linköping University, Sweden

Magnus Wahlström
Department of Computer Science, Royal Holloway, University of London, UK

Abstract

We study the parameterized complexity of MinCSP for so-called *equality languages*, i.e., for finite languages over an infinite domain such as $\mathbb{N}$, where the relations are defined via first-order formulas whose only predicate is $=$. This is an important class of languages that forms the starting point of all study of infinite-domain CSPs under the commonly used approach pioneered by Bodirsky, i.e., languages defined as reducts of finitely bounded homogeneous structures. Moreover, MinCSP over equality languages forms a natural class of optimisation problems in its own right, covering such problems as Edge Multicut, Steiner Multicut and (under singleton expansion) Edge Multiway Cut. We classify MinCSP(Γ) for every finite equality language Γ, under the natural parameter, as either FPT, W[1]-hard but admitting a constant-factor FPT-approximation, or not admitting a constant-factor FPT-approximation unless FPT=W[2]. In particular, we describe an FPT case that slightly generalises Multicut, and show a constant-factor FPT-approximation for Disjunctive Multicut, the generalisation of Multicut where the "cut requests" come as disjunctions over $O(1)$ individual cut requests $s_i \neq t_i$. We also consider *singleton expansions* of equality languages, enriching an equality language with the capability for assignment constraints $(x = i)$ for either a finite or infinitely many constants i, and fully characterize the complexity of the resulting MinCSP.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases parameterized complexity, constraint satisfaction, parameterized approximation

Digital Object Identifier 10.4230/LIPIcs.ESA.2023.86

Related Version *Full Version*: `https://arxiv.org/abs/2305.11131`

Funding *George Osipov*: Supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

1 Introduction

Let D be a fixed domain, and let Γ be a finite set of finitary relations over D. Γ is referred to as a *constraint language*. A *constraint* over Γ is a pair (R, X), less formally written $R(X)$, where $R \in \Gamma$ is a relation of some arity r and $X = (x_1, \ldots, x_r)$ is a tuple of variables. It is *satisfied* by an assignment α if $(\alpha(x_1), \ldots, \alpha(x_r)) \in R$. For a constraint language Γ, the *constraint satisfaction problem* over Γ, CSP(Γ), is the problem where an instance I is a collection of constraints over Γ, on some set of variables V, and the question is if there is an assignment satisfying all constraints in I. In the optimization variant MinCSP(Γ), the input also contains an integer k and the question is whether there is an assignment such that all but at most k constraints are satisfied. Less formally, a constraint language Γ determines the "type of constraints" allowed in an instance of CSP(Γ) or MinCSP(Γ), and varying the constraint language defines problems of varying complexity (such as k-SAT, k-Colouring, st-Min Cut, etc.). After decades-long investigations, *dichotomy theorems* have been established for these problems: for every constraint language over a finite domain, CSP(Γ) and MinCSP(Γ) is either in P or NP-complete, and the characterizations are known [16, 40, 38, 31]. For fixed cases, such as the Boolean domain $D = \{0, 1\}$, *parameterized* dichotomies are also known, characterizing every problem MinCSP(Γ) as either FPT or W[1]-hard [29], and similarly for approximate FPT algorithms [12]. This work represents significant advancements of our understanding of tractable and intractable computational problems (classical or parameterized).

But as highlighted by Bodirsky [3, 9], there are also many problems from a range of application domains that do not lend themselves to a formulation in the above CSP framework, yet which can be formulated via CSPs over structures with *infinite* domains. Unfortunately, CSPs with fixed templates over infinite domains are not as well-behaved as over finite domains; it is known that the problem CSP(Γ) over an infinite domain can have any computational complexity (including being intermediate), making any dichotomy impossible [6, 9]. There are also questions of how an arbitrary infinite-domain relation would be represented. The approach used by Bodirsky, which is the standard approach for the study of infinite-domain CSPs, is to consider a language Γ as a *reduct of a finitely bounded homogeneous structure.* Less technically, consider a structure, for example $(\mathbb{Q}, <)$ or $(\mathbb{Z}, <)$, and let Γ be a finite language where every relation in Γ has a quantifier-free first-order definition over the structure; i.e., Γ is a *first-order reduct* of the structure. For such languages a dichotomy is plausible, and many cases have been settled, including *temporal* CSPs, i.e., first-order reducts of $(\mathbb{Q}, <)$ [8]; *discrete temporal* CSPs, i.e., first-order reducts of $(\mathbb{Z}, <)$ [10]; CSPs over the universal random graph [11]; and many more.

Our goal is to study the parameterized complexity of MinCSPs over such structures. Many important problems in parameterized complexity, which are not well handled by CSP optimization frameworks over finite-domain CSPs, can be expressed very simply in this setting. For example, the MinCSP with domain $\mathbb{Q}$ and the single relation $<$ is equivalent to the Directed Feedback Arc Set problem, i.e., given a digraph D and an integer k, find a set X of at most k arcs from D such that $D - X$ is acyclic. (Here, the vertices of D become variables, the arcs constraints, and the topological order of $D - X$ becomes an assignment which violates at most $|X|$ constraints.) Other examples include Subset Directed Feedback Arc Set, which corresponds to MinCSP($<, \leq$), and Symmetric Directed Multicut which corresponds to MinCSP($\leq, \neq$). The former is another important FPT problem [18], while FPT status of the latter is open [23].

The structure we study in this paper is $(\mathbb{N}, =)$. The relations definable over this structure are called *equality constraint languages.* Here, $\mathbb{N}$ is an arbitrary, countably infinite domain; first-order reducts of $(\mathbb{N}, =)$ are simply relations definable by a quantifier-free first-order formula whose only predicate is $=$. Equivalently, relations in an equality language accept or reject an assignment to their arguments purely based on the partition that the assignment induces. Since every first-order formula is allowed to use equality in this framework, equality languages are contained in *every* other class of languages studied in the framework. Hence, characterizing the complexity of equality languages is a prerequisite for studying any other structure.

Moreover, the setting also covers problems that are important in their own right, as it captures undirected *graph separation* problems. In particular, (Vertex/Edge) Multicut is defined as follows. The input is a graph G, an integer k, and a set of *cut requests* $\mathcal{T} \subseteq \binom{V(G)}{2}$, and the task is to find a set X of at most k vertices, respectively edges, such that for every cut request $st \in \mathcal{T}$, vertices s and t are in different connected components in $G - X$. Multicut is FPT parameterized by k – a breakthrough result, settling a long-open question [37, 14]. As with the above examples, there appears to be no natural way of capturing Multicut as a finite-domain CSP optimization problem. However, it naturally corresponds to MinCSP$(=, \neq)$ over domain $\mathbb{N}$, where edges correspond to soft $=$-constraints and cut requests to crisp $\neq$-constraints. Another classic problem is Multiway Cut, which is the special case of Multicut where the cut requests are $\mathcal{T} = \binom{T}{2}$ for a set T of *terminal vertices* in the graph. Multiway Cut was among the first graph separation problems shown to be FPT [36], and remains a relevant problem, e.g., for the question of *polynomial kernelization* [32, 39]. While Multiway Cut is not directly captured by an equality CSP, it is captured by the *singleton expansion* of the setting, i.e., allowing "assignment constraints" of the form $x = i$ for all $i \in \mathbb{N}$.

Related work. Bodirsky and Kara [7] characterized CSP(Γ) as in P or NP-hard for every equality language Γ. Bodirsky, Chen and Pinsker [5] characterized the structure of equality languages up to pp-definitions (*primitive positive definitions*, see Section 2); these are too coarse to preserve the parameterized complexity of a problem, but their results are very useful as a guide to our search. For much more material on CSPs over infinite domains, see Bodirsky [4]. Singleton expansions (under different names) are discussed by Bodirsky [4] and Jonsson [25]. We have taken the term from Barto et al. [1].

Many variations on cut problems have been considered, and have been particularly important in parameterized complexity [20] (see also [19]). We cover Multicut, Multiway Cut and *Steiner cut*. Given a graph G and a set of terminals $T \subseteq V(G)$, a Steiner cut is an edge cut in G that separates T, i.e., a cut Z such that some pair of vertices in T is disconnected in $G - Z$. Steiner Cut is the problem of finding a minimum Steiner cut. This can clearly be solved in polynomial time; in fact, using advanced methods, it can even be computed in *near-linear* time [33, 17]. Steiner Multicut is the generalization where the input contains a set $\mathcal{T} = \{T_1, \ldots, T_t\}$ of terminal sets and the task is to find a smallest-possible cut that separates every set T_i. Since Multiway Cut with 3 terminals is NP-hard, Steiner Multicut is NP-hard if $t \geq 3$. Bringmann et al. [15] considered parameterized variations of this and showed, among other results, that Edge Steiner Multicut is FPT even for terminal sets T_i of unbounded size, if the parameter includes both t and the cut size k. On the other hand, parameterized by k alone, Steiner Multicut is W[1]-hard for terminal sets of size $|T_i| = 3$.

Other parameterized CSP dichotomies directly relevant to our work are the dichotomies for Boolean MinCSP as having constant factor FPT-approximations or being W[1]-hard to approximate [12] (with additional results in a later preprint [13]) and the recent FPT/W[1]-hardness dichotomy [29].

The area of FPT approximations has seen significant activity in recent years, especially regarding lower bounds on FPT approximations [24, 26, 2, 34]. In particular, we will need that there is no constant-factor FPT-approximation for Nearest Codeword in Boolean codes unless FPT=W[1] [13, 2], or for Hitting Set unless FPT=W[2] [34]. Lokshtanov et al. [35] considered fast FPT-approximations for problems whose FPT algorithms are slow; in particular, our result for Steiner Multicut builds on their algorithm giving an $O^*(2^{O(k)})$-time 2-approximation.

Our Results

We study the classical and parameterized complexity of MinCSP(Γ) for every finite equality language Γ, as well as for singleton expansions over equality languages. We consider both exact FPT-algorithms and constant-factor FPT-approximations, and for every finite Γ classify whether MinCSP(Γ) is in FPT, MinCSP(Γ) is W[1]-hard but admits constant-factor FPT-approximation, or MinCSP(Γ) is W[1]-hard to approximate within any constant. To describe the cases in more detail, we need some definitions.

Unsurprisingly, MinCSP(Γ) for an equality language Γ is NP-hard except in trivial cases, since MinCSP($=, \neq$) already corresponds to Edge Multicut. Specifically, MinCSP(Γ) is in P if Γ is *constant*, in which case every relation in Γ contains the tuple $(1, \ldots, 1)$, or *strictly negative*, in which case every relation in Γ contains every tuple $(1, \ldots, r)$ where all values are distinct (proper definitions of the terms are found in Section 3). In all other cases, MinCSP(Γ) is NP-hard by reduction from Edge Multicut. Moreover, under the Unique Games Conjecture [27], NP-hard MinCSP(Γ) does not admit polynomial-time constant-factor approximation.

To describe constraint languages Γ that give rise to fixed-parameter tractable MinCSP(Γ), let NEQ_3 be the ternary relation which contains all tuples with three distinct values, and let a *split* constraint be a constraint R of some arity $p + q$ for $p, q \geq 0$, defined (up to argument order) by

$$R(x_1, \ldots, x_p, y_1, \ldots, y_q) \equiv \bigwedge_{i,j \in [p]} (x_i = x_j) \wedge \bigwedge_{i \in [p], j \in [q]} (x_i \neq y_j).$$

We note that MinCSP(Γ) with split constraints reduces to Vertex Multicut. A split constraint $R(u_1, \ldots, u_p, v_1, \ldots, v_q)$ can be represented by introducing a new vertex c, adding edges cu_i for every $i \in [p]$ and cut requests cv_j for every $i \in [q]$. However, a constraint $\mathsf{NEQ}_3(u, v, w)$ cannot be handled by a gadget. Hence, we introduce the following auxiliary graph problem, and show that it is in FPT.

Vertex Multicut with Deletable Triples (aka Triple Multicut)

Instance:	A graph G, a collection $\mathcal{T} \subseteq \binom{V(G)}{3}$ of vertex triples, and integer k.
Parameter:	k.
Question:	Are there subsets $X_V \subseteq V(G)$ and $X_\mathcal{T} \subseteq \mathcal{T}$ such that $\lvert X_V \rvert + \lvert X_\mathcal{T} \rvert \leq k$ and every connected component of $G - X_V$ intersects every triple in $\mathcal{T} \setminus X_\mathcal{T}$ in at most one vertex?

MinCSP(Γ) for Γ with only split relations and NEQ_3 easily reduces to Triple Multicut. To complement this result with hardness, we prove the following.

▶ **Theorem 1.** *Let Γ be an equality constraint language that is neither constant nor strictly negative. Then MinCSP(Γ) is FPT if every relation in Γ is either split or* NEQ_3*, and W[1]-hard otherwise.*

Next, we describe the cases with constant-factor FPT-approximations. Consider relation

$$R_d = (x_1 \neq y_1 \vee \cdots \vee x_d \neq y_d)$$

and let Γ be a constraint language where every relation is defined by conjunction of relations R_d and $=$. We show that MinCSP(Γ) for such Γ is constant-factor fpt-approximable, with the factor depending on d. Again, we introduce a new graph problem to capture this case. Let G be a graph; a subset $L \subseteq \binom{V(G)}{2}$ of pairs is a *request list*, and a set of vertices $X \subseteq V(G)$ *satisfies* L if there is a pair $st \in L$ separated by X. For a graph G and a collection of request lists $\mathcal{L}$, let $\mathrm{cost}(G, \mathcal{L})$ be the minimum size of a set $X \subseteq V(G)$ that satisfies all lists in $\mathcal{L}$.

Disjunctive Multicut

Instance:	A graph G, a collection $\mathcal{L}$ of request lists, each of size at most d, and an integer k.
Parameter:	k.
Question:	Is $\mathrm{cost}(G, \mathcal{L}) \leq k$?

Our main algorithmic contribution is an FPT-approximation for Disjunctive Multicut. Note that Steiner Multicut is the special case of Disjunctive Multicut where each request list is $L = \binom{T_i}{2}$ for some terminal set T_i. Here we obtain an improved algorithm with approximation factor 2 and running time $O^*(2^{O(k)})$.

This precisely describes the FPT-approximable cases of MinCSP(Γ): For every equality constraint language Γ such that CSP(Γ) is in P, either MinCSP(Γ) reduces to Disjunctive Multicut in an immediate way (up to a constant-factor approximation loss), implying a constant-factor FPT-approximation, or there is a cost-preserving reduction from Hitting Set to MinCSP(Γ). We refer to the latter as MinCSP(Γ) being Hitting Set-hard.

▶ **Theorem 2.** *Let Γ be an equality constraint language such that CSP(Γ) is in P. Then either MinCSP(Γ) reduces to Disjunctive Multicut and has a constant-factor FPT-approximation, or MinCSP(Γ) is Hitting Set-hard.*

We can summarize the main result in the following way: for every finite equality constraint language Γ, either MinCSP(Γ) is FPT by reduction to Triple Multicut, MinCSP(Γ) is FPT-approximable by reduction to Disjunctive Multicut, or MinCSP(Γ) is Hitting Set-hard. A more technical description in terms of the allowed relations can be found in Section 3.

Singleton Expansion. In addition to the above (main) results, we also investigate the effect of adding constants to an equality language motivated by the problem Multiway Cut. More precisely, for an equality language Γ, we investigate the effect of adding some number of unary singleton relations $\{(i)\}$ to Γ. This is equivalent to allowing "assignment constraints" $(x = i)$ in MinCSP(Γ). We consider adding either a finite number of singletons, or every singleton relation. For an equality language Γ and an integer $c \in \mathbb{N}$, $c \geq 1$, we define $\Gamma_c^+ = \Gamma \cup \{\{(i) \mid i \in [c]\}$ as the language Γ with c different singletons added, and let Γ^+ denote Γ with every singleton $\{(i)\}$, $i \in \mathbb{N}$ added. Edge Multiway Cut corresponds to MinCSP(Γ^+) over the language $\Gamma = \{=\}$, and s-Edge Multiway Cut, the special case with s terminals, corresponds to MinCSP(Γ_s^+). By a *singleton expansion of* Γ we refer to either the language $\Gamma' = \Gamma^+$ or $\Gamma' = \Gamma_c^+$ for some $c \in \mathbb{N}$.

As the first step of the characterization, we observe that if Γ can express $=$ and $\neq$, then the singleton expansion adds no power, i.e., MinCSP(Γ^+) reduces back to MinCSP(Γ) by introducing variables $c_1, \ldots, c_m$ for arbitrarily many constants, adding constraints $c_i \neq c_j$ whenever $i \neq j$, and using constraints $x = c_i$ in place of assignments $x = i$. For the rest of the characterization, we thus study the cases that either cannot express equality, or cannot express disequality. In the former case, MinCSP(Γ') is always FPT and constant-factor approximable; the latter case is more involved. We defer the full case description to the full paper, but in summary, for any singleton expansion Γ' of a finite equality language Γ, we characterize MinCSP(Γ') as being in P or NP-hard, being FPT or W[1]-hard, and with respect to the existence of polynomial-time or FPT constant-factor approximations. Overall, the positive cases in the characterization follow without difficulty, but completing the picture with negative results requires significant additional work, building on the structural results of Bodirsky, Chen and Pinsker [5].

Roadmap. Section 2 contains technical preliminaries. Section 3 gives an overview of the classification proof. Section 4 shows the FPT algorithm for Triple Multicut. Section 5 gives the FPT approximation algorithms. This version omits several proofs, the approximation result for Steiner Multicut, and the classification of CSP and MinCSP for equality constraint languages under singleton expansion. These can be found in the full version.

2 Preliminaries

Graph Separation. Let G be an undirected graph. Denote the vertex set of G by $V(G)$ and the edge set by $E(G)$. For a subset of edges/vertices X in G, let $G - X$ denote the graph obtained by removing the elements of X from G, i.e. $G - X = (V(G), E(G) \setminus X)$ if $X \subseteq E(G)$ and $G - X = G[V(G) \setminus X]$ if $X \subseteq V(G)$. A *cut request* is a pair of vertices $st \in \binom{V(G)}{2}$, and an st-cut/st-separator is a subset of edges/vertices X such that $G - X$ contains no path connecting s and t. We write that X *fulfills* st if X is an st-cut/st-separator. We implicitly allows the inputs to cut problems such as Multiway Cut and Multicut to contain undeletable edges/vertices: for edges, with a parameter of k, we can include $k + 1$ parallel copies; for vertices, we can replace a vertex v with a clique of size $k + 1$, where every member of the clique has the same neighbourhood as v.

Parameterized Deletion. A *parameterized* problem is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is the input alphabet. The parameterized complexity class FPT contains problems decidable in $f(k) \cdot n^{O(1)}$ time, where f is a computable function and n is the bit-size of the instance. Let $L_1, L_2 \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. A mapping $F : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ is an *FPT-reduction* from L_1 to L_2 if

- $(x, k) \in L_1$ if and only if $F((x, k)) \in L_2$,
- the mapping can be computed in $f(k) \cdot n^{O(1)}$ time for some computable function f, and
- there is a computable function $g : \mathbb{N} \to \mathbb{N}$ such that for all $(x, k) \in \Sigma^* \times \mathbb{N}$, if $(x', k') = F((x, k))$, then $k' \le g(k)$.

The classes W[1] and W[2] contains all problems that are FPT-reducible to Clique and Hitting Set, respectively, parameterized by the solution size. These problems are not in FPT under the standard assumptions FPT≠W[1] and FPT≠W[2]. For a thorough treatment of parameterized complexity we refer to [20].

Constraint Satisfaction. Fix a *domain* D. A relation R of *arity* r is a subset of tuples in D^r, i.e. $R \subseteq D^r$. We write $=$ and $\neq$ to denote the binary equality and disequality relations over D, i.e. $\{(a, b) \in D^2 : a = b\}$ and $\{(a, b) \in D^2 : a \neq b\}$, respectively. A *constraint language* Γ is a set of relations over a domain D. A *constraint* is defined by a relation R and a tuple of variables $\mathbf{x} = (x_1, \dots, x_r)$, where r is the arity of R. It is often written as $R(\mathbf{x})$ or $R(x_1, \dots, x_r)$. An assignment $\alpha : \{x_1, \dots, x_r\} \to D$ *satisfies* the constraint if $\alpha(\mathbf{x}) = (\alpha(x_1), \dots, \alpha(x_r)) \in R$, and *violates* the constraint if $\alpha(\mathbf{x}) \notin R$.

Constraint Satisfaction Problem for Γ (CSP(Γ))

Instance:	An instance I, where $V(I)$ is a set of variables and $C(I)$ is a multiset of constraints using relations from Γ.
Question:	Is there an assignment $\alpha : V(I) \to D$ that satisfies all constraints in $C(I)$?

MinCSP is an optimization version of the problem seeking an assignment that minimizes the number of violated constraints. In this constraints are allowed to be *crisp* and *soft*. The *cost of assignment* α in an instance I of CSP is infinite if it violates a crisp constraint, and equals the number of violated soft constraints otherwise. The *cost of an instance* I denoted by $\mathrm{cost}(I)$ is the minimum cost of any assignment to I.

MinCSP(Γ)

Instance:	An instance I of CSP(Γ) and an integer k.
Question:	Is $\mathrm{cost}(I) \le k$?

Next, we recall a useful notion that captures local reductions between CSPs.

▶ **Definition 3.** *Let Γ be a constraint language over D and $R \subseteq D^r$ be a relation. A* primitive positive definition (pp-definition) *of R in Γ is an instance C_R of CSP$(\Gamma, =)$ with primary variables $\mathbf{x}$, auxiliary variables $\mathbf{y}$ and the following properties:*
(1) *if α satisfies C_R, then it satisfies $R(\mathbf{x})$,*
(2) *if α satisfies $R(\mathbf{x})$, then there exists an extension of α to $\mathbf{y}$ that satisfies C_R.*

Informally, pp-definitions can be used to simulate R using the relations available in Γ and equality: every constraint using R can be replaced by a gadget based on the pp-definition, resulting in an equivalent instance. The type of reductions captured by pp-definitions is however incompatible with MinCSP because the reductions do not preserve assignment costs. This motivates the following definition.

▶ **Definition 4.** *Let Γ be a constraint language over D and $R \subseteq D^r$ be a relation. An* implementation *of R in Γ is a pp-definition of R with primary variables $\mathbf{x}$, auxiliary variables $\mathbf{y}$ and an additional property: if α violates $R(\mathbf{x})$, there exists an extension of α to $\mathbf{y}$ of cost one.*

Although pp-definitions do not preserve costs, they can be used to simulate crisp constraints in MinCSP instances.

▶ **Proposition 5** (Proposition 5.2 in [30]). *Let Γ be a constraint language over a domain D and R be a relation over D. Then the following hold.*
1. *If Γ pp-defines R, then there is an FPT-reduction from MinCSP(Γ, R) restricted to instances with only crisp R-constraints to MinCSP$(\Gamma, =)$.*
2. *If Γ implements R, there is an FPT-reduction from MinCSP(Γ, R) to MinCSP$(\Gamma, =)$.*

Approximation. A minimization problem over an alphabet Σ is a triple $(\mathcal{I}, \mathrm{sol}, \mathrm{cost})$, where $\mathcal{I} \subseteq \Sigma^*$ is the set of instances, $\mathrm{sol} : \mathcal{I} \to \Sigma^*$ is a function such that maps instances $I \in \mathcal{I}$ to the sets of solutions $\mathrm{sol}(I)$, and $\mathrm{cost} : \mathcal{I} \times \Sigma^* \to \mathbb{Z}_{\geq 0}$ is a function that takes an instance $I \in \mathcal{I}$ and a solution $X \in \mathrm{sol}(I)$ as input, and returns a non-negative integer cost of the solution. Define $\mathrm{cost}(I) := \min\{\mathrm{cost}(I, X) : X \in \mathrm{cost}(I)\}$. A constant-factor approximation algorithm with factor $c \geq 1$ takes an instance $x \in \mathcal{I}$ and an integer $k \in \mathbb{N}$, and returns "yes" if $\mathrm{cost}(I) \leq k$ and "no" if $\mathrm{cost}(I) > c \cdot k$. A *cost-preserving reduction* from a problem $A = (\mathcal{I}_A, \mathrm{sol}_A, \mathrm{cost}_A)$ to $B = (\mathcal{I}_B, \mathrm{sol}_B, \mathrm{cost}_B)$ is a pair of polynomial-time computable functions F and G such that (1) for every $I \in \mathcal{I}_A$, we have $F(I) \in \mathcal{I}_B$ with $\mathrm{cost}_A(I) = \mathrm{cost}_B(F(I))$, and (2) for every $I \in \mathcal{I}_A$ and $Y \in \mathrm{sol}(F(I))$, we have $G(I, Y) \in \mathrm{sol}(I)$, and $\mathrm{cost}_A(I, G(I, Y)) \leq \mathrm{cost}_B(F(I), Y)$. If there is a cost-preserving reduction from A to B, and B admits a constant-factor polynomial-time/fpt approximation algorithm, then A also admits a constant-factor polynomial-time/fpt approximation algorithm.

3 Classification Overview

We now give an overview of the complexity dichotomy. Details are deferred to the full paper. We begin with a definition of the relevant language classes. Recall that an *equality language* is a constraint language over $\mathbb{N}$ whose relations can be defined via Boolean formulas over the equality predicate. More precisely, for a set of variables $X = \{x_1, \ldots, x_n\}$, let a *positive literal* be a term $(x_i = x_j)$ and a *negative literal* a term $(x_i \neq x_j)$, $i, j \in [n]$. A *clause* is a disjunction of literals. Then every equality relation has a CNF definition as a conjunction of clauses. A relation (respectively language) is *Horn* if it (respectively every relation in the

Table 1 Selected Horn relations R and the complexity of MinCSP$(R, =, \neq)$. FPA refers to fixed-parameter approximation.

Name	CNF Formula	Tuples	Complexity
EQ_3	$(x_1 = x_2) \wedge (x_2 = x_3) \wedge (x_1 = x_3)$	$(1,1,1)$	FPT
NEQ_3	$(x_1 \neq x_2) \wedge (x_2 \neq x_3) \wedge (x_1 \neq x_3)$	$(1,2,3)$	FPT
—	$(x_1 = x_2) \wedge (x_1 \neq x_3) \wedge (x_2 \neq x_3)$	$(1,1,2)$	FPT
ODD_3	$(x_1 = x_2 \vee x_1 \neq x_3) \wedge (x_1 = x_2 \vee x_2 \neq x_3) \wedge (x_1 \neq x_2 \vee x_2 \neq x_3)$	$(1,1,1), (1,2,3)$	Hitting Set-hard
NAE_3	$(x_1 \neq x_2 \vee x_2 \neq x_3)$	excludes $(1,1,1)$	W[1]-hard, FPA
$R^{\vee}_{\neq,\neq}$	$(x_1 \neq x_2 \vee x_3 \neq x_4) \wedge (x_1 \neq x_3) \wedge (x_1 \neq x_4) \wedge (x_2 \neq x_3) \wedge (x_2 \neq x_4)$	$(1,2,3,3)$, $(1,1,2,3)$, $(1,2,3,4)$	W[1]-hard, FPA
$R^{\wedge}_{=,=}$	$(x_1 = x_2) \wedge (x_3 = x_4)$	$(1,1,1,1)$, $(1,1,2,2)$	W[1]-hard, FPA
$R^{\wedge}_{\neq,\neq}$	$(x_1 \neq x_2) \wedge (x_3 \neq x_4)$	– *too many too list here* –	W[1]-hard, FPA
$R^{\wedge}_{=,\neq}$	$(x_1 = x_2) \wedge (x_3 \neq x_4)$	$(1,1,1,2), (1,1,2,1), (1,1,2,3)$	W[1]-hard, FPA

language) has a CNF definition where every clause has at most one positive literal, *negative* if positive literals only occur in singleton clauses $(x_i = x_j)$, *strictly negative* if there are no positive literals, and *conjunctive* if all clauses are singletons. Note that split relations and NEQ_3 are both conjunctive.

Bodirsky and Kara [7] showed that for an equality language Γ, CSP(Γ) is in P if Γ is Horn or constant, and NP-hard otherwise. We note that MinCSP(Γ) is trivial if Γ is constant or strictly negative, and also show that if Γ is Horn but not constant or strictly negative then Γ implements the relations $=$ and $\neq$. Hence we focus on this case and assume that Γ is Horn and $=, \neq \in \Gamma$. The polynomial-time complexity classification then follows since Edge Multicut reduces to MinCSP$(=, \neq)$. For the remaining steps, we show that

1. MinCSP$(\Gamma, =, \neq)$ admits a constant-factor FPT-approximation if Γ is negative, otherwise it is Hitting Set-hard;
2. MinCSP$(R, =, \neq)$ is W[1]-hard if R is negative but not conjunctive;
3. MinCSP$(R, =, \neq)$ is W[1]-hard if R is conjunctive but neither split nor NEQ_3;
4. MinCSP$(\Gamma, =, \neq)$ is in FPT if Γ is conjunctive and all relations in Γ are split or NEQ_3.

Table 1 lists a number of Horn relations and the associated complexity of MinCSP.

Towards hardness of approximation, we recall a result of Bodirsky, Chen and Pinsker [5] that every equality language that is not negative can define ODD_3 (see Table 1). We show that MinCSP$(\mathsf{ODD}_3, =, \neq)$ is Hitting Set-hard.

▶ **Lemma 6.** *There is a cost-preserving reduction from Hitting Set to MinCSP$(\mathsf{ODD}_3, =, \neq)$ where every ODD_3-constraint is crisp.*

Proof Sketch. Let the input be $(V, \mathcal{E}, k)$, $V = \{1, \ldots, n\}$. Create an instance (I, k) of MinCSP$(\mathsf{ODD}_3, =, \neq)$ starting from variables $x_1, \ldots, x_n$ and z, with soft constraints $x_i = z$ for all $i \in [n]$. For every set $e = \{a_1, \ldots, a_\ell\} \in \mathcal{E}$, add auxiliary variables $y_2, \ldots, y_\ell$ and crisp constraints $\mathsf{ODD}_3(x_{a_1}, x_{a_2}, y_2)$, $\mathsf{ODD}_3(y_{i-1}, x_{a_i}, y_i)$ for all $3 \leq i \leq \ell$, and $x_{a_1} \neq y_\ell$. These constraints are satisfiable if and only if not all variables x_i, $i \in e$ are equal, so to satisfy them it is sufficient to break a soft constraint $x_{a_i} = z$. Thus, $X \subseteq V$ is a hitting set if and only if $I - \{x_i = z : i \in X\}$ is consistent. ◀

As noted, this implies that MinCSP(Γ) is W[1]-hard to even approximate in FPT time. Now assume that Γ is negative. Then every relation $R \in \Gamma$ is defined by a conjunction of positive singleton clauses and strictly negative clauses. For a constant-factor approximation,

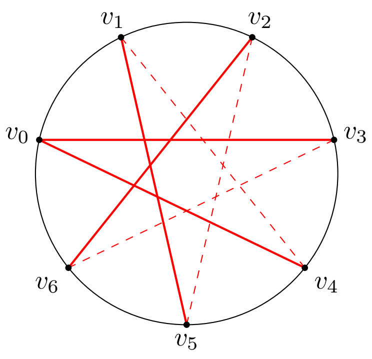

Figure 1 An illustration of a choice gadget for $t = 3$. Black arcs correspond to equality constraints, dashed red edges to soft disequality constraints, and the bold red edge to a crisp disequality constraint.

by [12, Lemma 10] we may split these definitions into separate constraints, so we may assume that an instance of MinCSP(Γ) uses constraint types $(x_i = x_j)$ and $(x_1 \neq y_1 \vee \ldots \vee x_r \neq y_r)$, of lengths $r \leq d$ for some constant d. Then MinCSP(Γ) reduces to Disjunctive Multicut, and since $d = O(1)$ we get an FPT approximation. This settles the cases where MinCSP(Γ) has a constant-factor FPT approximation.

Towards delineating FPT and W[1]-hard cases, assume that Γ is negative but not conjunctive, and let $R \in \Gamma$ be a relation such that every CNF definition of R contains a non-singleton clause. We show that in this case $\{R, =, \neq\}$ pp-defines $R^{\vee}_{\neq,\neq}$ or NAE_3, and that MinCSP($R', =, \neq$) is W[1]-hard with crisp R'-constraints, where R' is either $R^{\vee}_{\neq,\neq}$ or NAE_3. The problem MinCSP($\mathsf{NAE}_3, =$) with crisp NAE_3-constraints naturally corresponds to Steiner Multicut, which is W[1]-hard [15]. For the remaining proofs, we will need a *choice gadget.* Let $S = \{s_1, \ldots, s_t\}$ be a set. Define an instance $W(S)$ of CSP($=, \neq$) as follows. Introduce $2t + 1$ variables $v_0, \ldots, v_{2t}$ and identify indices modulo $2t + 1$, e.g. $v_0 = v_{2t+1}$. Connect variables in a cycle of equalities, i.e. add soft constraints $v_i = v_{i+1}$ for all $0 \leq i \leq 2t$. The *forward partner* of a variable v_i is $f(v_i) := v_{i+t}$, i.e. the variable that is t steps ahead of v_i on the cycle. Add soft constraints $v_i \neq f(v_i)$ for all $0 \leq i \leq 2t$, and make the constraint $v_0 \neq v_t$ crisp. See Figure 1 for an illustration.

▶ **Lemma 7.** *Let S be a set of size at least two and $W(S)$ be the choice gadget. Then* $\text{cost}(W(S)) = 3$. *Moreover, if $X \subseteq W(S)$, $|X| = 3$, $W(S) - X$ is consistent and X contains $v_i \neq f(v_i)$, then X also contains $v_{i-1} = v_i$ and $f(v_i) = f(v_{i+1})$.*

Proof Sketch. Since $v_0 \neq v_t$ is a crisp constraint, we need to remove one link from the top and one from the the bottom chain of equality constraints connecting v_0 and v_t. After this deletion, at least one chain of length $\lceil \frac{2t+1-2}{2} \rceil = t$ remains, which connects a vertex v_i with its forward partner $f(v_i)$. Thus, we either need to disconnect v_i and $f(v_i)$, or remove the constraint $v_i \neq f(v_i)$. Observe that it is sufficient to delete $v_{i-1} = v_i$, $f(v_i) = f(v_{i+1})$ and $v_i \neq f(v_i)$ to satisfy $W(S)$. Moreover, if a deletion set X of size 3 contains $v_i \neq f(v_i)$, the only pair of vertices v_j and $f(v_j)$ that may remain connected in $W(S) - X$ is the pair with $j = i$. Out of the two compatible choices, only deleting $v_{i-1} = v_i$ and $f(v_i) = f(v_{i+1})$ leaves no constraint unsatisfied. ◀

Intuitively, deleting $v_{i-1} = v_i$, $f(v_i) = f(v_{i+1})$ and $v_i \neq f(v_i)$ from $W(S)$ corresponds to choosing element s_i from the set S. We note a simple reduction from Multicoloured Independent Set to MinCSP($R^{\vee}_{\neq,\neq}, =$). Let the input be a graph G with k colour classes $V(G) = V_1 \cup \ldots V_k$. Create a choice gadget for every V_i without soft $\neq$-constraints (but keeping the crisp one), and set the budget to $2k$. For every pair $u \in V_i$, $v \in V_j$ for $i \neq j$ such that $uv \in E(G)$, add a crisp constraint $R^{\vee}_{\neq,\neq}(u, f(u), v, f(v))$. Due to the budget, $u = f(u)$ holds for at least one $u \in V_i$ for every V_i, corresponding to a selection, and the R-constraint prevents that $u = f(u)$ and $v = f(v)$, thereby blocking the combination of $u \in V_i$ and $v \in V_j$. We defer the details.

Having shown that MinCSP$(\Gamma, =, \neq)$ is W[1]-hard if Γ is non-conjunctive, it remains to show hardness for languages which are conjunctive but not NEQ_3 or split. We use the following W[1]-hard problem (see [22, Lemma 6.1]).

Split Paired Cut

Instance:	Graphs G_1, G_2, vertices $s_1, t_1 \in V(G_1)$, $s_2, t_2 \in V(G_2)$, a family of disjoint edge pairs $\mathcal{P} \subseteq E(G_1) \times E(G_2)$, and an integer k.
Parameter:	k.
Question:	Is there a subset $X \subseteq \mathcal{P}$ of size at most k such that for both $i \in \{1, 2\}$, $\{e_i : \{e_1, e_2\} \in X\}$ is an st-cut in G_i?

Assuming Γ is conjunctive, associate with a relation $R(x_1, \dots, x_r)$ a graph where for each pair $i, j \in [r]$ there is a blue edge $x_i x_j$ if $i \neq j$ and R implies $x_i = x_j$, and a red edge $x_i x_j$ if R implies $x_i \neq x_j$. Then the graph of a split relation $R(X, Y)$, $X = \{x_1, \dots, x_p\}$ and $Y = \{y_1, \dots, y_q\}$ is a blue clique on X and a red biclique of (X, Y), and the graph of NEQ_3 is a red triangle. The remaining cases are precisely the cases when the graph contains two disjoint edges $x_1 x_2$, $x_3 x_4$ with no blue edges connecting their endpoints (such as $R^{\wedge}_{=,=}$, $R^{\wedge}_{=,\neq}$ or $R^{\wedge}_{\neq,\neq}$). There is a curious parallel to the characterization of FPT cases of Boolean MinCSP, in terms of 2CNF-definable relations whose *Gaifman graph* is $2K_2$-free [29]. We show hardness for all such cases. If both $x_1 x_2$ and $x_3 x_4$ are blue, a reduction is immediate from Split Paired Cut. We sketch the reduction for a more interesting case of MinCSP$(R^{\wedge}_{\neq,\neq}, =, \neq)$, and note that the reduction for MinCSP$(R^{\wedge}_{=,\neq}, =, \neq)$ is a variant of the above, and hence omitted in this version of the paper.

▶ **Lemma 8.** *MinCSP$(R^{\wedge}_{\neq,\neq}, =)$ is W[1]-hard.*

Proof sketch. Let $(G_1, G_2, s_1, t_1, s_2, t_2, \mathcal{P}, k)$ be an instance of Split Paired Cut. By the construction of [28, Lemma 5.7], assume $k = 2\ell$, and $\mathcal{F}_i$ for $i \in \{1, 2\}$ are $s_i t_i$-maxflows in G_i partitioning $E(G_i)$ into k pairwise edge-disjoint paths. Construct an instance (I, k') of MinCSP$(R, =, \neq)$ with $k' = 5\ell$ as follows. Start by creating a variable for every vertex in $V(G_1) \cup V(G_2)$ with the same name as the vertex. For each $i \in \{1, 2\}$, consider a path $P \in \mathcal{F}_i$, and let p be the number of edges on P. Create a choice gadget $W(P)$ for every P with variables $v^P_0, \dots, v^P_p$ following the path, and fresh variables v^P_j for $p < j \leq 2p$ added to the instance. Observe that variables may appear on several paths in $\mathcal{F}_i$. In particular, $v^P_0 = s_i$ and $v^P_p = t_i$ for every $P \in \mathcal{F}_i$, so we have crisp constraints $s_i \neq t_i$. Furthermore, since $\mathcal{F}_i$ partitions $E(G_i)$, the construction contains a copy of graphs G_1 and G_2 with equality constraints for edges. Now we pair up edges according to $\mathcal{P}$. For every pair $\{e_1, e_2\} \in \mathcal{P}$, let $P \in \mathcal{F}_1$ and $Q \in \mathcal{F}_2$ be the paths such that $e_1 \in P$ and $e_2 \in Q$, and suppose $e_1 = v^P_{i-1} v^P_i$ and $e_2 = v^Q_{j-1} v^Q_j$. Pair up soft constraints $v^P_i \neq f(v^P_i)$ and $v^Q_j \neq f(v^Q_j)$, i.e. replace individual constraints with one soft constraint $R(v^P_i, f(v^P), v^Q_j, f(v^Q_j))$. Finally, if an edge $uv \in E(G)$ does not appear in any pair of $\mathcal{P}$, make constraint $u = v$ crisp in I. This completes the construction.

The proof of correctness is deferred to the full paper. ◀

For all remaining cases, as noted, every $R \in \Gamma$ is either split or NEQ_3, and there is a simple reduction from MinCSP(Γ) to Triple Multicut. Since the latter is FPT (Theorem 10) the classification is complete.

We omit the details of the classification for singleton expansion cases. The proofs are somewhat more intricate, explicitly using the algebraic method as employed by Bodirsky, Chen and Pinsker [5], but ultimately both the hardness proofs and algorithmic cases are less interesting than for the above.

4 Triple Multicut

We show that Triple Multicut is in FPT, thus proving Theorem 10. The algorithm works by a reduction to Boolean MinCSP, i.e. MinCSP(Δ) for a constraint language Δ over the binary domain $\{0, 1\}$. Parameterized complexity of Boolean MinCSP was completely classified by [29]. As a result of our reduction, we obtain an instance where Δ is bijunctive, i.e. every relation in Δ can be defined by a Boolean formula in CNF with at most two literals in each clause. Define the *Gaifman graph* of a bijunctive relation R with vertices $\{1, \dots, r\}$, where r is the arity of R, and edges ij for every pair of indices such that $R(x_1, \dots, x_r)$ implies a 2-clause involving x_i and x_j. A graph is $2K_2$-free if no four vertices induce a subgraph with two independent edges.

▶ **Theorem 9** (Theorem 1.2 of [29]). *Let Δ be a finite bijunctive Boolean constraint language such that the Gaifman graphs of all relations in Δ are $2K_2$-free. Then MinCSP(Δ) is in FPT.*

We are ready to present the algorithm.

▶ **Theorem 10.** *Triple Multicut is fixed-parameter tractable.*

Proof Sketch. Let $(G, \mathcal{T}, k)$ be an instance of Triple Multicut. By iterative compression, we obtain $X_V \subseteq V(G)$ and $X_\mathcal{T} \subseteq \mathcal{T}$ such that $|X_V| + |X_\mathcal{T}| \leq k + 1$ and all components of $G - X_V$ intersect triples in $\mathcal{T} \setminus X_\mathcal{T}$ in at most one vertex. Moreover, by branching on the intersection, we can assume that a hypothetical optimal solution $(Z_V, Z_\mathcal{T})$ to $(G, \mathcal{T}, k)$ is disjoint from $(X_V, X_\mathcal{T})$. Let $X = X_V \cup \bigcup_{uvw \in X_\mathcal{T}} \{u, v, w\}$ and guess the partition of the vertices in X into connected components of $G - Z_V$. Identify vertices that belong to the same component, and enumerate them via the bijective mapping $\alpha : X \to \{1, \dots, d\}$. Observe that for every triple $uvw \in X_\mathcal{T}$, values $\alpha(u)$, $\alpha(v)$ and $\alpha(w)$ are distinct since $X_\mathcal{T} \cap Z_\mathcal{T} = \emptyset$. Create an instance I_α of Boolean MinCSP as follows.

1. Introduce variables v_i and $\hat{v}_i$ for every $v \in V(G)$ and $i \in [d]$.
2. For every vertex $v \in V(G)$, add soft constraint $\bigwedge_{i<j}(\neg v_i \vee \neg v_j) \wedge \bigwedge_i (v_i \to \hat{v}_i)$.
3. For every vertex $v \in X$, add crisp constraints $v_{\alpha(v)}$, $\hat{v}_{\alpha(v)}$, and $\neg v_j$, $\neg \hat{v}_j$ for all $j \neq \alpha(v)$.
4. For every edge $uv \in E(G)$ and $i \in [d]$, add crisp constraints $\hat{u}_i \to v_i$ and $\hat{v}_i \to u_i$.
5. For every triple $uvw \in \mathcal{T}$ and $i \in [d]$, add soft constraints $(\neg \hat{u}_i \vee \neg \hat{v}_i) \wedge (\neg \hat{v}_i \vee \neg \hat{w}_i) \wedge (\neg \hat{u}_i \vee \neg \hat{w}_i)$.

This completes the reduction. We defer the correctness proof to the full version. ◀

5 Disjunctive Multicut

We show that Disjunctive Multicut is constant-factor fpt-approximable. Section 5 presents the main loop of the Disjunctive Multicut algorithm, while Section 5 is dedicated to the most technical subroutine of the algorithm that involves *randomized covering of shadow* [37].

Main Loop of the Disjunctive Multicut Algorithm

Let G be a graph with vertices $V(G) = V^\infty(G) \uplus V^1(G)$ partitioned into undeletable and deletable, respectively. A subset $L \subseteq \binom{V(G)}{2}$ of pairs is a *request list*, and a set of vertices $X \subseteq V(G)$ *satisfies* L if there is a pair $st \in L$ separated by X. This includes the possibility that $s \in X$ or $t \in X$. For a graph G and a collection of request lists $\mathcal{L}$, we let $\operatorname{cost}(G, \mathcal{L})$ be the minimum size of a set $X \subseteq V^1(G)$ that satisfies all lists in $\mathcal{L}$. Disjunctive Multicut asks, given an instance $(G, \mathcal{L})$, whether $\operatorname{cost}(G, \mathcal{L}) \leq k$.

Disjunctive Multicut problem generalizes not only Multicut (which is a special case with $d = 1$) but also d-Hitting Set. To see the latter, take an edgeless graph G and make every request a *singleton*, i.e. a pair ss for a vertex $s \in V(G)$. The only way to satisfy a singleton ss is to delete the vertex s itself, and the only way to satisfy a list of singletons is to delete one of the vertices in it.

The intuitive idea behind the approximation algorithm for Disjunctive Multicut is to iteratively simplify the instance $(G, \mathcal{L}, k)$, making it closer to Bounded Hitting Set after each iteration. Roughly, we make progress if the maximum number of non-singleton requests in a list decreases. In each iteration, the goal is to find a set of $O(k)$ vertices whose deletion, combined with some branching steps, simplifies every request list. This process can continue for $O(d)$ steps until we obtain an instance of Bounded Hitting Set, which can be solved in fpt time by branching. The instance may increase in the process, but finally we obtain a solution of cost $f(d) \cdot k$ for some function f. We do not optimize for f in our proofs. Observe also that in the context of constant-factor fpt approximability, some dependence on d is unavoidable since the problem with unbounded d generalizes Hitting Set.

Formally, for a request list L, let $\mu_1(L)$ and $\mu_2(L)$ be the number of singleton and non-singleton cut requests in L, respectively. Define the measure for a list L as $\mu(L) = \mu_1(L) + 3\mu_2(L) = |L| + 2\mu_2(L)$, and extend it to a collection of list requests $\mathcal{L}$ by taking the maximum, i.e. $\mu(\mathcal{L}) = \max_{L \in \mathcal{L}} \mu(L)$. Observe that $\mu(\mathcal{L}) \leq 3d$ for any instance of Disjunctive Multicut. Further, let $V(L) = \bigcup_{st \in L} \{s, t\}$ denote the set of vertices in a list L, and $\nu(\mathcal{L}) = \mu_1(L) + 2\mu_2(L)$ be an upper bound on the maximum number of variable occurrences in a list of $\mathcal{L}$. The workhorse of the approximation algorithm is the following lemma.

▶ **Lemma 11.** *There is a randomized algorithm* Simplify *that takes an instance $(G, \mathcal{L}, k)$ of* Disjunctive Multicut *as input, and in $O^*(2^{O(k)})$ time produces a graph G' and a collection of requests $\mathcal{L}'$ such that $|V(G')| \leq |V(G)|$, $\nu(\mathcal{L}') \leq \nu(\mathcal{L})$, $|\mathcal{L}'| \leq k^2|\mathcal{L}|$, and $\mu(\mathcal{L}') \leq \mu(\mathcal{L}) - 1$. Moreover, the following holds.*

- *If $\operatorname{cost}(G, \mathcal{L}) \leq k$, then, with probability $2^{-O(k^2)}$, we have $\operatorname{cost}(G', \mathcal{L}') \leq 2k$.*
- *If $\operatorname{cost}(G, \mathcal{L}) > 3k$, then we have $\operatorname{cost}(G', \mathcal{L}') > 2k$.*

Randomization in Lemma 11 comes from the use of the *random covering of shadow* of [37, 18]. They also provide a derandomized version of this procedure, so our algorithm can be derandomized as well. We postpone the proof of Lemma 11 until Section 5 since it requires introduction of some technical machinery. For now, we show how to prove Theorem 12 using the result of the lemma.

▶ **Theorem 12.** *Disjunctive Multicut is fixed-parameter tractable.*

Proof. Let $(G, \mathcal{L}, k)$ be an instance of Disjunctive Multicut. Repeat the following steps until $\mu_2(\mathcal{L}) = 0$. Apply the algorithm of Lemma 11 to $(G, \mathcal{L}, k)$, obtaining a new graph G and a new collection of lists $\mathcal{L}$, and let $(G, \mathcal{L}, k) := (G', \mathcal{L}', 2k)$. When $\mu_2(\mathcal{L}) = 0$, let

Algorithm 1 Main Loop.

```
1: procedure SolveDJMC($G, \mathcal{L}, k$)
2:     while $\mu_2(\mathcal{L}) > 0$ do
3:         $(G, \mathcal{L}) \leftarrow$ Simplify($G, \mathcal{L}, k$)
4:         if Simplify rejects then
5:             reject
6:         $k \leftarrow 2k$
7:     $W \leftarrow \{vv : v \in V(G)\}$
8:     if SolveHittingSet($W, \mathcal{L}, k$) accepts then
9:         accept
10:    else
11:        reject
```

$W = \{vv : v \in V(G)\}$ be the set of singleton cut requests for every vertex in $V(G)$. Check whether $(W, \mathcal{L}, k)$ is a yes-instance of Hitting Set – if yes, accept, otherwise reject. See Algorithm 1 for the pseudocode.

To argue correctness, let $(G, \mathcal{L}, k)$ be the input instance and $(G', \mathcal{L}', k')$ be the instance obtained after simplification. By induction and Lemma 11, we have $|V(G')| \leq |V(G)|$ and $\nu(\mathcal{L}') \leq \nu(\mathcal{L})$. Since $\nu(\mathcal{L}') \leq \nu(\mathcal{L}) \leq 2d$ and $\mu_2(\mathcal{L}) = 0$, every list in $\mathcal{L}$ has at most $2d$ requests. Let r be the number of calls to Simplify performed by the algorithm. Note that $r \leq \mu(\mathcal{L}) \leq 3d$ since the measure decreases by at least one with each iteration, and define $k' = 2^r k$. The lists in $\mathcal{L}'$ only contain singletons, thus $(G', \mathcal{L}', k')$ is essentially an instance of Hitting Set with sets of size $2d$. Moreover, $|\mathcal{L}'| \leq k^{2r}|\mathcal{L}|$, so the number of lists is polynomial in $|\mathcal{L}|$. We can solve $(G', \mathcal{L}', k')$ in $O^*((2d)^{k'})$ time by branching (see, for example, Chapter 3 in [20]). For the other direction, suppose $\text{cost}(G, \mathcal{L}) \leq k$. By Lemma 11 and induction, we have $\text{cost}(G', \mathcal{L}') \leq 2^r k \leq k'$ with probability $2^{-O(rk^2)}$, and the algorithm accepts. If $\text{cost}(I) > 3k$, then $\text{cost}(G', \mathcal{L}') > k'$ and the algorithm rejects. ◀

Simplification Procedure

In this section we prove Lemma 11. We start by iterative compression and guessing. Then we delete at most k vertices from the graph and modify it, obtaining an instance amenable to the main technical tool of the section – the *shadow covering* technique.

Initial Phase

Let $(G, \mathcal{L}, k)$ be an instance of Disjunctive Multicut. By iterative compression, assume we have a set $X \subseteq V(G)$ that satisfies all lists in $\mathcal{L}$ and $|X| = c \cdot k + 1$, where $c := c(d)$ is the approximation factor. Assume Z is an optimal solution to G, i.e. $|Z| \leq k$ and Z satisfies all lists in $\mathcal{L}$. Guess the intersection $W = X \cap Z$, and let $G' = G - W$, $X' = X \setminus W$, and $Z' = Z \setminus W$. Construct $\mathcal{L}'$ starting with $\mathcal{L}$ and removing all lists satisfied by W. Further, guess the partition $\mathcal{X} = (X_1, \ldots, X_\ell)$ of X' into the connected components of $G' - Z'$, and identify the variables in each subset X_i into a single vertex x_i, and redefine X' accordingly. Note that the probability of our guesses being correct up to this point is $2^{-O(k \log k)}$. Also, these steps can be derandomized by creating $2^{O(k \log k)}$ branches.

Now compute a minimum $\mathcal{X}$-multiway cut in G', i.e. a set $M \subseteq V^1(G')$ that separates every pair of vertices x_i and x_j in X'. Note that Z' is a $\mathcal{X}$-multiway cut by the definition of $\mathcal{X}$, so $|M| \leq |Z'| \leq k$. Such a set M can be computed in $O^*(2^k)$ time using the algorithm of [21]. If no $\mathcal{X}$-multiway cut of size at most k exists, then abort the branch and make another guess for $\mathcal{X}$. If an $\mathcal{X}$-multiway cut M of size at most k is obtained, remove the vertices in M from G and along with the lists in $\mathcal{L}'$ satisfied by M. This completes the initial phase of the algorithm. Properties of the resulting instance are summarized below.

▶ **Lemma 13** (Proof Omitted). *After the initial phase we obtain a graph G', a family of list requests $\mathcal{L}'$, and subset of vertices $X' \subseteq V(G')$ such that $|V(G')| \leq |V(G)|$, $\nu(\mathcal{L}') \leq \nu(\mathcal{L})$, $\mu(\mathcal{L}') \leq \mu(\mathcal{L})$, and $|X'| \in O(k)$. The set X' satisfies all lists in $\mathcal{L}'$ and intersects each connected component of G' in at most one vertex. Moreover, the following hold.*

- $\mathrm{cost}(G, \mathcal{L}) \leq k + \mathrm{cost}(G', \mathcal{L}')$.
- *If* $\mathrm{cost}(G, \mathcal{L}) \leq k$*, then, with probability* $2^{-O(k \log k)}$*, we have* $\mathrm{cost}(G', \mathcal{L}') \leq k$*. Moreover, there is a set $Z' \subseteq V(G')$, $|Z'| \leq k$ that satisfies all lists in $\mathcal{L}'$ and is disjoint from X'.*

Random Covering of Shadow

Random covering of shadow is a powerful tool introduced by [37] and sharpened by [18]. We use the latter work as our starting point. Although [18] present their theorems in terms of directed graphs, their results are applicable to our setting by considering undirected edges as bidirectional, i.e. replacing every edge uv with a pair of antiparallel arcs (u, v) and (v, u). Consider a graph G with vertices partitioned into deltable and undeletable subsets, i.e. $V(G) = V^1(G) \uplus V^\infty(G)$. Let $\mathcal{F} = (F_1, \ldots, F_q)$ be a family of connected subgraphs of G. An *$\mathcal{F}$-transversal* is a set of vertices T that intersects every subgraph F_i in $\mathcal{F}$. If T is an $\mathcal{F}$-transversal, we say that $\mathcal{F}$ is *T-connected.* For every $W \subseteq V(G)$, the *shadow of W (with respect to T)* is the subset of vertices disconnected from T in $G - W$. We state it for the case $T \subseteq V^\infty(G)$ which suffices for our applications.

▶ **Theorem 14** (Random Covering of Shadow, Theorem 3.5 in [18]). *There is an algorithm* RandomCover *that takes a graph G, a subset $T \subseteq V^\infty(G)$ and an integer k as input, and in $O^*(4^k)$ time outputs a set $S \subseteq V(G)$ such that the following holds. For any family $\mathcal{F}$ of T-connected subgraphs, if there is an $\mathcal{F}$-transversal of size at most k in $V^1(G)$, then with probability $2^{-O(k^2)}$, there exists an $\mathcal{F}$-transversal $Y \subseteq V^1(G)$ of size at most k such that*
1. *$Y \cap S = \emptyset$, and*
2. *S covers the shadow of Y with respect to T.*

The following consequence is convenient for our purposes.

▶ **Corollary 15** (Proof Omitted). *Let S and Y be the shadow-covering set and the $\mathcal{F}$-transversal from Theorem 14, respectively. Define $R = V(G) \setminus S$ to be the complement of S. Then $Y \subseteq R$ and, for every vertex $v \in R$, either $v \in Y$ or v is connected to T in $G - Y$.*

Note that if a vertex $v \in N(S)$ and v is undeletable, then $v \in R$ and $v \notin Y$, hence v is connected to T in $G - Y$. Since $Y \cap S = \emptyset$, every vertex in $N(v) \cap S$ is also connected to T in $G - Y$, so we can remove $N(v) \cap S$ from S (and add it to R instead). By applying this procedure to exhaustion, we may assume that no vertex in $N(S)$ is undeletable.

With the random covering of shadow at our disposal, we return to Disjunctive Multicut. By Lemma 13, we can start with an instance $(G, \mathcal{L}, k)$ and a set $X \subseteq V(G)$ such that $|X| \in O(k)$, X satisfies all lists in $\mathcal{L}$, every connected component of G intersects X in at most one vertex, and there is an optimal solution Z disjoint from X. Let $\mathcal{T} := \mathcal{T}(G, \mathcal{L}, X, Z)$ be the set of cut requests in $\bigcup \mathcal{L}$ satisfied by both X and Z. Define $\mathcal{F}$ as the set of st-walks for all $st \in \mathcal{T}$. Observe that an $\mathcal{F}$-transversal is precisely a $\mathcal{T}$-multicut. Apply the algorithm from Theorem 14 to (G, X, k). Since X and Z are $\mathcal{F}$-transversals and $|Z| \leq k$ by assumption, Theorem 14 and Corollary 15 imply that we can obtain a set $R \subseteq V(G)$ in fpt time such that, with probability $2^{-O(k^2)}$, there is an $\mathcal{F}$-transversal $Y \subseteq R$ of size at most k, and every vertex in $R \setminus Y$ is connected to X in $G - Y$.

For every vertex $v \in V^1(G) \setminus X$, define a set of vertices $R_v \subseteq R \setminus X$ as follows:

- if v is disconnected from X, then let $R_v = \emptyset$;
- if $v \in N(X)$ or $v \in R$, then let $R_v = \{v\}$;
- otherwise, let $R_v = R \cap N(H)$, where H is the component of $G[S]$ containing v.

Note that, by definition, the set R_v is an Xv-separator in G. Moreover, we have ensured that $N(S)$ does not contain undeletable vertices, so R_v does not contain undeletable vertices. In a certain sense, the sets R_v are the only Xv-separators that Y needs to use. This idea is made precise in the following lemma.

▶ **Lemma 16** (Proof Omitted). *Let G be a graph. Let X and Y be disjoint subsets of $V(G)$ such that X intersects every connected component of G in at most one vertex. Suppose $R \subseteq V(G)$ is such that $Y \subseteq R$ and all vertices in $R \setminus Y$ are connected to X in $G - Y$. If a vertex s is disconnected from X in $G - Y$, then $R_s \subseteq Y$.*

Now we compute a simplified collection of lists $\mathcal{L}'$. Start by adding all lists in $\mathcal{L}$ to $\mathcal{L}'$. Remove every singleton request xx such that $x \in X$ from every list of $\mathcal{L}'$. For every list $L \in \mathcal{L}'$ not shortened this way, let $st \in L$ be a non-singleton cut request satisfied by X. Consider R_s and R_t and apply one of the following rules.

(R1) If $|R_s| > k$ and $|R_t| > k$, remove st from L.
(R2) If $|R_s| \leq k$ and $|R_t| > k$, replace L with sets $(L \setminus \{st\}) \cup \{aa\}$ for all $a \in R_s$.
(R3) If $|R_s| > k$ and $|R_t| \leq k$, replace L with sets $(L \setminus \{st\}) \cup \{bb\}$ for all $b \in R_t$.
(R4) If $|R_s| \leq k$ and $|R_t| \leq k$, replace L with sets $(L \setminus \{st\}) \cup \{aa, bb\}$ for all $a \in R_t$, $b \in R_t$.

Finally, make vertices in X undeletable, obtaining a new graph G'. This completes the simplification step. Note that each list in $\mathcal{L}$ is processed once, so the running time of the last step is polynomial.

Now we prove some properties of G' and $\mathcal{L}'$ obtained above. Note that $|V(G')| = |V(G)|$. Since every list in $\mathcal{L}$ is processed once and with at most k^2 new lists, the size of $|\mathcal{L}'|$ grows by a factor of at most k^2. To see that $\nu(\mathcal{L}') \leq \nu(\mathcal{L})$ and $\mu(\mathcal{L}') \leq \mu(\mathcal{L}) - 1$, observe that every reduction rule replaces a list L with new lists with either one less non-singleton request (so μ_2 decreases by at least 1), and adds up to two singleton requests (so μ_1 increases by at most 2). Moreover, in every list of $\mathcal{L}$ there is a cut request satisfied by X, so no list of $\mathcal{L}$ remains unchanged in $\mathcal{L}'$. We state the remaining ingredients for proving correctness.

▶ **Lemma 17** (Proof Omitted). *If $\operatorname{cost}(G, \mathcal{L}) \leq k$, then, with probability $2^{-O(k^2)}$, we have $\operatorname{cost}(G', \mathcal{L}') \leq 2k$.*

▶ **Lemma 18** (Proof Omitted). *If $\operatorname{cost}(G, \mathcal{L}) > 2k$, then we have $\operatorname{cost}(G', \mathcal{L}') > 2k$.*

We are now ready to prove Lemma 11.

Proof of Lemma 11. Suppose $(G, \mathcal{L}, k)$ is a yes-instance of Disjunctive Multicut. By Lemma 13, after the initial phase we obtain G', $\mathcal{L}'$ such that $|V(G')| \leq |V(G)|$, $\nu(\mathcal{L}') \leq \nu(\mathcal{L})$, $\mu(\mathcal{L}') \leq \mu(\mathcal{L})$, and $\operatorname{cost}(G', \mathcal{L}') \leq k$. Moreover, we obtain a set $X' \subseteq V(G')$, $|X'| \in O(k)$, that satisfies all lists in $\mathcal{L}'$, intersects every component of G' in at most one vertex, and is disjoint from an optimum solution Z' to $(G', \mathcal{L}', k)$. Now we apply random covering of shadow and the list reduction rules to $G', \mathcal{L}', X'$, obtaining a new graph G'' and a new set of lists $\mathcal{L}''$. By Lemma 17, with probability $2^{O(-k^2)}$, we have $\operatorname{cost}(G'', \mathcal{L}'') \leq 2k$. This proves one statement of Lemma 11. For the second statement of Lemma 11, suppose $\operatorname{cost}(G'', \mathcal{L}'') \leq 2k$. By Lemma 18, we have $\operatorname{cost}(G', \mathcal{L}') \leq 2k$. By Lemma 13, $\operatorname{cost}(G'\mathcal{L}') \leq 2k$ implies that $\operatorname{cost}(G, \mathcal{L}) \leq 2k + k \leq 3k$, and we are done. ◀

References

1 Libor Barto, Andrei A. Krokhin, and Ross Willard. Polymorphisms, and how to use them. In *The Constraint Satisfaction Problem*, volume 7 of *Dagstuhl Follow-Ups*, pages 1–44. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

2 Arnab Bhattacharyya, Édouard Bonnet, László Egri, Suprovat Ghoshal, Karthik C. S., Bingkai Lin, Pasin Manurangsi, and Dániel Marx. Parameterized intractability of even set and shortest vector problem. *J. ACM*, 68(3):16:1–16:40, 2021.

3 Manuel Bodirsky. *Constraint satisfaction with infinite domains*. PhD thesis, Humboldt University of Berlin, Unter den Linden, Germany, 2004.

4 Manuel Bodirsky. *Complexity of Infinite-Domain Constraint Satisfaction*. Lecture Notes in Logic (LNL). Cambridge University Press, 2021. doi:10.1017/9781107337534.

5 Manuel Bodirsky, Hubie Chen, and Michael Pinsker. The reducts of equality up to primitive positive interdefinability. *J. Symb. Log.*, 75(4):1249–1292, 2010.

6 Manuel Bodirsky and Martin Grohe. Non-dichotomies in constraint satisfaction complexity. In *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 184–196. Springer, 2008.

7 Manuel Bodirsky and Jan Kára. The complexity of equality constraint languages. *Theory of Computing Systems*, 43(2):136–158, 2008.

8 Manuel Bodirsky and Jan Kára. The complexity of temporal constraint satisfaction problems. *J. ACM*, 57(2):9:1–9:41, 2010.

9 Manuel Bodirsky and Marcello Mamino. Constraint satisfaction problems over numeric domains. In *The Constraint Satisfaction Problem*, volume 7 of *Dagstuhl Follow-Ups*, pages 79–111. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

10 Manuel Bodirsky, Barnaby Martin, and Antoine Mottet. Discrete temporal constraint satisfaction problems. *J. ACM*, 65(2):9:1–9:41, 2018.

11 Manuel Bodirsky and Michael Pinsker. Schaefer's theorem for graphs. *J. ACM*, 62(3):19:1–19:52, 2015.

12 Édouard Bonnet, László Egri, and Dániel Marx. Fixed-parameter approximability of Boolean MinCSPs. In *24th Annual European Symposium on Algorithms (ESA 2016)*, pages 18:1–18:18, 2016.

13 Édouard Bonnet, László Egri, Bingkai Lin, and Dániel Marx. Fixed-parameter approximability of boolean MinCSPs, 2018. doi:10.48550/arXiv.1601.04935.

14 Nicolas Bousquet, Jean Daligault, and Stéphan Thomassé. Multicut is FPT. *SIAM J. Comput.*, 47(1):166–207, 2018. doi:10.1137/140961808.

15 Karl Bringmann, Danny Hermelin, Matthias Mnich, and Erik Jan Van Leeuwen. Parameterized complexity dichotomy for Steiner multicut. *Journal of Computer and System Sciences*, 82(6):1020–1043, 2016.

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

17 Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 – November 3, 2022*, pages 612–623. IEEE, 2022. doi:10.1109/FOCS54457.2022.00064.

18 Rajesh Chitnis, Marek Cygan, Mohammataghi Hajiaghayi, and Dániel Marx. Directed subset feedback vertex set is fixed-parameter tractable. *ACM Transactions on Algorithms (TALG)*, 11(4):1–28, 2015.

19 Christophe Crespelle, Pål Grønås Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification, 2020. arXiv:2001.06867.

20 Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*. Springer, 2015.

21 Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. On multiway cut parameterized above lower bounds. *ACM Transactions on Computation Theory (TOCT)*, 5(1):1–11, 2013.

22 Konrad K Dabrowski, Peter Jonsson, Sebastian Ordyniak, George Osipov, and Magnus Wahlström. Almost consistent systems of linear equations. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3179–3217. SIAM, 2023.

23 Eduard Eiben, Clément Rambaud, and Magnus Wahlström. On the parameterized complexity of symmetric directed multicut. In *IPEC*, LIPIcs. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

24 Andreas Emil Feldmann, Karthik C. S., Euiwoong Lee, and Pasin Manurangsi. A survey on approximation in parameterized complexity: Hardness and algorithms. *Algorithms*, 13(6):146, 2020.

25 Peter Jonsson. Constants and finite unary relations in qualitative constraint reasoning. *Artif. Intell.*, 257:1–23, 2018.

26 Karthik C. S. and Subhash Khot. Almost polynomial factor inapproximability for parameterized k-clique. In *CCC*, volume 234 of *LIPIcs*, pages 6:1–6:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

27 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775, 2002.

28 Eun Jung Kim, Stefan Kratsch, Marcin Pilipczuk, and Magnus Wahlström. Solving hard cut problems via flow-augmentation. *arXiv e-prints*, pages arXiv–2007, 2020.

29 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. In *SODA*, pages 3218–3228. SIAM, 2023.

30 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. *arXiv preprint*, 2023. arXiv:2207.07422.

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

32 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *Journal of the ACM (JACM)*, 67(3):1–50, 2020.

33 Jason Li and Debmalya Panigrahi. Deterministic min-cut in poly-logarithmic max-flows. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 85–92. IEEE, 2020. doi:10.1109/FOCS46700.2020.00017.

34 Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. Constant approximating parameterized k-SETCOVER is W[2]-hard. In *SODA*, pages 3305–3316. SIAM, 2023.

35 Daniel Lokshtanov, Pranabendu Misra, MS Ramanujan, Saket Saurabh, and Meirav Zehavi. FPT-approximation for FPT problems. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 199–218. SIAM, 2021.

36 Dániel Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006. doi:10.1016/j.tcs.2005.10.007.

37 Daniel Marx and Igor Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM Journal on Computing*, 43(2):355, 2014.

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

39 Magnus Wahlström. Quasipolynomial multicut-mimicking networks and kernels for multiway cut problems. *ACM Trans. Algorithms*, 18(2):15:1–15:19, 2022.

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