On the Parameterized Complexity of Diverse SAT

Neeldhara Misra
Department of Computer Science and Engineering, Indian Institute of Technology Gandhinagar, India

Harshil Mittal
Department of Computer Science and Engineering,
Indian Institute of Technology Gandhinagar, India

Ashutosh Rai
Department of Mathematics, Indian Institute of Technology Delhi,
New Delhi, India

Abstract

We study the Boolean Satisfiability problem (SAT) in the framework of diversity, where one asks for multiple solutions that are mutually far apart (i.e., sufficiently dissimilar from each other) for a suitable notion of distance/dissimilarity between solutions. Interpreting assignments as bit vectors, we take their Hamming distance to quantify dissimilarity, and we focus on the problem of finding two solutions. Specifically, we define the problem Max Differ SAT (resp. Exact Differ SAT) as follows: Given a Boolean formula ϕ on n variables, decide whether ϕ has two satisfying assignments that differ on at least (resp. exactly) d variables. We study the classical and parameterized (in parameters d and $n-d$) complexities of Max Differ SAT and Exact Differ SAT, when restricted to some classes of formulas on which SAT is known to be polynomial-time solvable. In particular, we consider affine formulas, Krom formulas (i.e., 2-CNF formulas) and hitting formulas.

For affine formulas, we show the following: Both problems are polynomial-time solvable when each equation has at most two variables. Exact Differ SAT is NP-hard, even when each equation has at most three variables and each variable appears in at most four equations. Also, Max Differ SAT is NP-hard, even when each equation has at most four variables. Both problems are W[1]-hard in the parameter $n-d$. In contrast, when parameterized by d, Exact Differ SAT is W[1]-hard, but Max Differ SAT admits a single-exponential FPT algorithm and a polynomial-kernel. For Krom formulas, we show the following: Both problems are polynomial-time solvable when each variable appears in at most two clauses. Also, both problems are W[1]-hard in the parameter d (and therefore, it turns out, also NP-hard), even on monotone inputs (i.e., formulas with no negative literals). Finally, for hitting formulas, we show that both problems can be solved in polynomial-time.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Diverse solutions, Affine formulas, 2-CNF formulas, Hitting formulas

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.50

Funding *Neeldhara Misra*: Supported by IIT Gandhinagar and the SERB ECR grant ECR/2018/002967.
Harshil Mittal: Supported by IIT Gandhinagar and Google as an ISAAC sponsor.
Ashutosh Rai: Supported by IIT Delhi.

Acknowledgements We thank Saraswati Girish Nanoti for helpful discussions.

1 Introduction

We initiate a study of the problem of finding two satisfying assignments to an instance of SAT, with the goal of maximizing the number of variables that have different truth values under the two assignments, in the parameterized setting. This question is motivated by the broader framework of finding "diverse solutions" to optimization problems. When a real-world problem is modelled as a computational problem, some contextual side-information is often lost. So, while two solutions may be equally good for the theoretical formulation, one of them may be better than the other for the actual practical application. A natural fix is to provide multiple solutions (instead of just one solution) to the user, who may then pick the solution that best fulfills her/his need. However, if the solutions so provided are all quite similar to each other, they may exhibit almost identical behaviours when judged on the basis of any relevant external factor. Thus, to ensure that the user is able to meaningfully compare the given solutions and hand-pick one of them, she/he must be provided a collection of *diverse solutions*, i.e., a few solutions that are sufficiently dissimilar from each other. This framework of *diversity* was proposed by Baste et. al. [3]. Since the late 2010s, several graph-theoretic and matching problems have been studied in this setting from an algorithmic standpoint. These include diverse variants of vertex cover [4], feedback vertex set [4], hitting set [4], perfect/maximum matching [17], stable matching [20], weighted basis of matroid [18], weighted common independent set of matroids [18], minimum s-t cut [11], spanning tree [22] and non-crossing matching [33].

The BOOLEAN SATISFIABILITY PROBLEM (SAT) asks whether a given Boolean formula has a satisfying assignment. This problem serves a crucial role in complexity theory [27], cryptography [32] and artificial intelligence [37]. In the early 1970s, SAT became the first problem proved to be NP-complete in independent works of Cook [8] and Levin [30]. Around the same time, Karp [27] built upon this result by showing NP-completeness of twenty-one graph-theoretic and combinatorial problems via reductions from SAT. In the late 1970s, Schaefer [35] formulated the closely related GENERALIZED SATISFIABILITY PROBLEM (SAT(S)), where each constraint applies on some variables, and it forces the corresponding tuple of their truth-values to belong to a certain Boolean relation from a fixed finite set S. His celebrated dichotomy result listed six conditions such that SAT(S) is polynomial-time solvable if S meets one of them; otherwise, SAT(S) is NP-complete.

Since SAT is NP-complete, it is unlikely to admit a polynomial-time algorithm, unless $P = NP$. Further, in the late 1990s, Impaglliazo and Paturi [25] conjectured that SAT is unlikely to admit even sub-exponential time algorithms, often referred to as the *exponential-time hypothesis*. To cope with the widely believed hardness of SAT, several special classes of Boolean formulas have been identified for which SAT is polynomial-time solvable. In the late 1960s, Krom [29] devised a quadratic-time algorithm to solve SAT on 2-CNF formulas. In the late 1970s, follow-up works of Even et. al. [16] and Aspvall et. al. [2] proposed linear-time algorithms to solve SAT on 2-CNF formulas. These algorithms used limited back-tracking and analysis of the strongly-connected components of the implication graph respectively. In the late 1980s, Iwama [26] introduced the class of hitting formulas, for which he gave a closed-form expression to count the number of satisfying assignments in polynomial-time. It is also known that SAT can be solved in polynomial-time on affine formulas using Gaussian elimination [21]. Some other polynomial-time recognizable classes of formulas for which SAT is polynomial-time solvable include Horn formulas [12, 36], CC-balanced formulas [7], matched formulas [19], renamable-Horn formulas [31] and q-Horn DNF formulas [5, 6].

Diverse variant of SAT. In this paper, we undertake a complexity-theoretic study of SAT in the framework of diversity. We focus on the problem of finding a *diverse pair of satisfying assignments* of a given Boolean formula, and we take the number of variables on which the two assignments differ as a measure of dissimilarity between them. Specifically, we define the problem MAX DIFFER SAT (resp. EXACT DIFFER SAT) as follows: Given a Boolean formula ϕ on n variables and a non-negative integer d, decide whether there are two satisfying assignments of ϕ that differ on at least d (resp. exactly d) variables. That is,

Table 1 Classical and parameterized (in parameters d and $n - d$) complexities of EXACT DIFFER SAT, when restricted to affine formulas, 2-CNF formulas and hitting formulas.

	Classical complexity	Parameter d	Parameter $n - d$
Affine formulas	NP-hard, even on $(3, 4)$-affine formulas (Theorem 1) Polynomial-time on 2-affine formulas (Theorem 3)	W[1]-hard (Theorem 4)	W[1]-hard (Theorem 7)
2-CNF formulas	Polynomial-time on $(2, 2)$-CNF formulas (Theorem 9)	W[1]-hard (Theorem 10)	?
Hitting formulas	Polynomial-time (Theorem 11)	–	–

Table 2 Classical and parameterized (in parameters d and $n - d$) complexities of MAX DIFFER SAT, when restricted to affine formulas, 2-CNF formulas and hitting formulas.

	Classical complexity	Parameter d	Parameter $n - d$
Affine formulas	NP-hard, even on 4-affine formulas (Theorem 2) Polynomial-time on 2-affine formulas (Theorem 3)	Single-exponential FPT (Theorem 5) Polynomial kernel (Theorem 6)	W[1]-hard (Theorem 7)
2-CNF formulas	Polynomial-time on $(2, 2)$-CNF formulas (Theorem 8)	W[1]-hard (Theorem 10)	?
Hitting formulas	Polynomial-time (Theorem 11)	–	–

this problem asks whether there are two satisfying assignments of ϕ that overlap on at most $n - d$ (resp. exactly $n - d$) variables. Note that SAT can be reduced to its diverse variant by setting d to 0. Thus, as SAT is NP-hard in general, so is MAX/EXACT DIFFER SAT. So, it is natural to study the diverse variant on those classes of formulas for which SAT is polynomial-time solvable. In particular, we consider affine formulas, 2-CNF formulas and hitting formulas. We refer to the corresponding restrictions of MAX/EXACT DIFFER SAT as MAX/EXACT DIFFER AFFINE-SAT, MAX/EXACT DIFFER 2-SAT and MAX/EXACT DIFFER HITTING-SAT respectively. We analyze the classical and parameterized (in parameters d and $n - d$) complexities of these problems.

Related work. This paper is not the first one to study algorithms to determine the maximum number of variables on which two solutions of a given SAT instance can differ. Several exact exponential-time algorithms are known to find a pair of maximally far-apart satisfying assignments. In the mid 2000s, Angelsmark and Thapper [1] devised an $\mathcal{O}(1.7338^n)$ time algorithm to solve MAX HAMMING DISTANCE 2-SAT. Their algorithm involved a careful analysis of the micro-structure graph and used a solver for weighted 2-SAT as a subroutine. Around the same time, Dahlöff [10] proposed an $\mathcal{O}(1.8348^n)$ time algorithm for MAX HAMMING DISTANCE XSAT. In the late 2010s, follow-up works of Hoi et. al. [24, 23] developed algorithms for the same problem with improved running times, i.e., $\mathcal{O}(1.4983^n)$ for the general case, and $\mathcal{O}(1.328^n)$ for the case when every clause has at most three literals.

Parameterized complexity. In the 1990s, Downey and Fellows [14] laid the foundations of parameterized algorithmics. This framework measures the running time of an algorithm as a function of both the input size and a *parameter* k, i.e., a suitably chosen attribute of the input. Such a fine-grained analysis helps to cope with the lack of polynomial-time algorithms for NP-hard problems by instead looking for an algorithm with running time whose super-polynomial explosion is confined to the parameter k alone. That is, such an algorithm has a running time of the form $f(k) \cdot n^{\mathcal{O}(1)}$, where $f(\cdot)$ is any computable function (could be exponential, or even worse) and n denotes the input size. Such an algorithm is said to be *fixed-parameter tractable* (FPT) because its running time is polynomially-bounded for every fixed value of the parameter k. For more on this paradigm, see [9].

Our findings. We summarize our findings in Table 1 and Table 2. In Section 3, we show that

- Exact Differ Affine-SAT is NP-hard, even on $(3, 4)$-affine formulas,
- Max Differ Affine-SAT is NP-hard, even on 4-affine formulas,
- Exact/Max Differ Affine-SAT is polynomial-time solvable on 2-affine formulas,
- Exact Differ Affine-SAT is W[1]-hard in the parameter d,
- Max Differ Affine-SAT admits a single-exponential FPT algorithm in the parameter d,
- Max Differ Affine-SAT admits a polynomial kernel in the parameter d, and
- Exact/Max Differ Affine-SAT is W[1]-hard in the parameter $n - d$.

In Section 4, we show that Exact/Max Differ 2-SAT can be solved in polynomial-time on $(2, 2)$-CNF formulas, and Exact/Max Differ 2-SAT is W[1]-hard in the parameter d. In Section 5, we show that Exact/Max Differ Hitting-SAT is polynomial-time solvable.

2 Preliminaries

A Boolean variable can take one of the two truth values: 0 (False) and 1 (True). We use n to denote the number of variables in a Boolean formula ϕ. An *assignment* of ϕ is a mapping from the set of all its n variables to $\{0, 1\}$. A *satisfying assignment* of ϕ is an assignment σ such that ϕ evaluates to 1 under σ, i.e., when every variable x is substituted with its assigned truth value $\sigma(x)$. We say that two assignments σ_1 and σ_2 *differ* on a variable x if they assign different truth values to x. That is, one of them sets x to 0, and the other sets x to 1. Otherwise, we say that σ_1 and σ_2 *overlap* on x. That is, either both of them set x to 0, or both of them set x to 1.

A *literal* is either a variable x (called a *positive literal*) or its negation $\neg x$ (called a *negative literal*). A *clause* is a disjunction (denoted by $\bigvee$) of literals. A Boolean formula in *conjunctive normal form*, i.e., a conjunction (denoted by $\bigwedge$) of clauses, is called a *CNF formula*. A 2-*CNF formula* is a CNF formula with at most two literals per clause. A $(2, 2)$-*CNF formula* is a 2-CNF formula in which each variable appears in at most two clauses. An *affine formula* is a conjunction of linear equations over the two-element field $\mathbb{F}_2$. We use $\oplus$ to denote the XOR operator, i.e., addition-modulo-2. A 2-*affine formula* is an affine formula in which each equation has at most two variables. Similarly, a 3-*affine* (resp. 4-*affine*) *formula* is an affine formula in which each equation has at most three (resp. four) variables. A $(3, 4)$-*affine formula* is a 3-affine formula in which each variable appears in at most four equations.

The solution set of a system of linear equations can be obtained in polynomial-time using Gaussian elimination [21]. It may have no solution, a unique solution or multiple solutions. When it has multiple solutions, the solution set is described as follows: Some variables are allowed to take any value; we call them *free variables*. The remaining variables take values that are dependent on the values taken by the free variables; we call them *forced variables*. That is, the value taken by any forced variable is a linear combination of the values taken by some free variables. For example, consider the following system of three linear equations over

$\mathbb{F}_2$: $x \oplus y \oplus z = 1$, $u \oplus y = 1, w \oplus z = 1$. This system has multiple solutions, and its solution set can be described as $\{(x, y, z, u, w) \mid y \in \mathbb{F}_2, z \in \mathbb{F}_2, x = y \oplus z \oplus 1, u = y \oplus 1, w = z \oplus 1\}$. Here, y and z are free variables. The remaining variables, i.e., x, u and w, are forced variables.

A *hitting formula* is a CNF formula such that for any pair of its clauses, there is some variable that appears as a positive literal in one clause, and as a negative literal in the other clause. That is, no two of its clauses can be simultaneously falsified. Note that the number of unsatisfying assignments of a hitting formula ϕ on n variables can be expressed as follows:

$$\sum_{C\colon C \text{ is a clause of } \phi} |\{\sigma \mid \sigma \text{ is an assignment of } \phi \text{ that falsifies } C\}| = \sum_{C\colon C \text{ is a clause of } \phi} 2^{n-|vars(C)|}$$

Here, we use $vars(C)$ to denote the set of all variables that appear in the clause C.

We use the following as source problems in our reductions:

- Independent Set. Given a graph G and a positive integer k, decide whether G has an independent set of size k. This problem is known to be NP-hard on cubic graphs [34], and W[1]-hard in the parameter k [15].
- Multicolored Clique. Given a graph G whose vertex set is partitioned into k color-classes, decide whether G has a k-sized clique that picks exactly one vertex from each color-class. This problem is known to be NP-hard on r-regular graphs [9].
- Exact Even Set. Given a universe $\mathcal{U}$, a family $\mathcal{F}$ of subsets of $\mathcal{U}$ and a positive integer k, decide whether there is a set $X \subseteq \mathcal{U}$ of size exactly k such that $|X \cap S|$ is even for all sets S in the family $\mathcal{F}$. This problem is known to be W[1]-hard in the parameter k [13].
- Odd Set (resp. Exact Odd Set). Given a universe $\mathcal{U}$, a family $\mathcal{F}$ of subsets of $\mathcal{U}$ and a positive integer k, decide whether there is a set $X \subseteq \mathcal{U}$ of size at most k (resp. exactly k) such that $|X \cap S|$ is odd for all sets S in the family $\mathcal{F}$. Both these problems are known to be W[1]-hard in the parameter k [13].

We use a polynomial-time algorithm for the following problem as a sub-routine:

- Subset Sum problem. Given a multi-set of integers $\{w_1, \ldots, w_p\}$ and a target sum k, decide whether there exists $X \subseteq [p]$ such that $\sum_{i \in X} w_i = k$. This problem is known to be polynomial-time solvable when the input integers are specified in unary [28].

We use the notation $\mathcal{O}^\star(\cdot)$ to hide polynomial factors in running time.

3 Affine formulas

In this section, we focus on Exact Differ Affine-SAT, i.e, finding two different solutions to affine formulas. To begin with, we show that finding two solutions that differ on *exactly* d variables is hard even for $(3, 4)$-affine formulas: recall that these are instances where every equation has at most three variables and every variable appears in at most four equations.

▶ **Theorem 1.** *Exact Differ Affine-SAT is* NP*-hard, even on* $(3, 4)$*-affine formulas.*

Proof. We describe a reduction from Independent Set on Cubic graphs. Consider an instance (G, k) of Independent Set, where G is a cubic graph. We construct an affine formula ϕ as follows: For every vertex $v \in V(G)$, introduce a variable x_v, its $3k$ copies (say $x_v^1, \ldots, x_v^{3k}$), and $3k$ equations: $x_v \oplus x_v^1 = 0$, $x_v^1 \oplus x_v^2 = 0, \ldots, x_v^{3k-1} \oplus x_v^{3k} = 0$. For every edge $e = uv \in E(G)$, introduce variable y_e and equation $x_u \oplus x_v \oplus y_e = 0$. We set $d = k \cdot (3k + 4)$. For every vertex $v \in V(G)$, the variable x_v appears in four equations (i.e., $x_v \oplus x_v^1 = 0$ and the three equations corresponding to the three edges incident to v in G), each of $x_v^1, \ldots, x_v^{3k-1}$ appears in two equations, and x_v^{3k} appears in one equation. For every edge $e \in E(G)$, the variable y_e appears in one equation. So, overall, every variable appears in at most four equations. Also, the equation corresponding to any edge contains three variables, and the remaining equations contain two variables each. Therefore, ϕ is a $(3, 4)$-affine formula.

Now, we prove that (G, k) is a YES instance of Independent Set if and only if (ϕ, d) is a YES instance of Exact Differ Affine-SAT. At a high level, we argue this equivalence as follows: In the forward direction, we show that the two desired satisfying assignments are the all 0 assignment, and the assignment that i) assigns 1 to every x variable (and also, its $3k$ copies) that corresponds to a vertex of the independent set, ii) assigns 1 to every y variable that corresponds to an edge that has one endpoint inside the independent set and the other endpoint outside it, iii) assigns 0 to every x variable (and also, its $3k$ copies) that corresponds to a vertex outside the independent set, and iv) assigns 0 to every y variable that corresponds to an edge that has both its endpoints outside the independent set. In the reverse direction, we show that the desired k-sized independent set consists of those vertices that correspond to the x variables on which the two assignments differ. We now turn to a proof of equivalence.

Forward direction. Suppose that G has a k-sized independent set, say S. Let σ_1 and σ_2 be assignments of ϕ defined as follows: For every vertex $v \in V(G) \setminus S$, both σ_1 and σ_2 set $x_v, x_v^1, \ldots, x_v^{3k}$ to 0. For every vertex $v \in S$, σ_1 sets $x_v, x_v^1, \ldots, x_v^{3k}$ to 0, and σ_2 sets $x_v, x_v^1, \ldots, x_v^{3k}$ to 1. For every edge $e \in E(G)$ that has both its endpoints in $V(G) \setminus S$, both σ_1 and σ_2 set y_e to 0. For every edge $e \in E(G)$ that has one endpoint in S and the other endpoint in $V(G) \setminus S$, σ_1 sets y_e to 0, and σ_2 sets y_e to 1.

As σ_1 sets all variables to 0, it is clear that it satisfies ϕ. Now, we show that σ_2 satisfies ϕ. Consider any edge $e = uv \in E(G)$ and its corresponding equation $x_u \oplus x_v \oplus y_e = 0$. If both endpoints of e belong to $V(G) \setminus S$, then σ_2 sets x_u, x_v and y_e to 0. Also, if e has one endpoint (say u) in S, and the other endpoint in $V(G) \setminus S$, then σ_2 sets x_u to 1, x_v to 0 and y_e to 1. Therefore, in both cases, $x_u \oplus x_v \oplus y_e$ takes the truth value 0 under σ_2. Also, for any vertex $v \in V(G)$, since σ_2 gives the same truth value to $x_v, x_v^1, \ldots, x_v^{3k}$ (i.e., all 1 if $v \in S$, and all 0 if $v \in V(G) \setminus S$), it also satisfies the equations $x_v \oplus x_v^1 = 0, x_v^1 \oplus x_v^2 = 0, \ldots, x_v^{3k-1} \oplus x_v^{3k} = 0$. Thus, σ_2 is a satisfying assignment of ϕ.

As G is a cubic graph, every vertex in S is incident to three edges in G. Also, as S is an independent set, none of these edges has both endpoints in S. Therefore, there are $3 \cdot |S|$ edges that have one endpoint in S and the other endpoint in $V(G) \setminus S$. Note that σ_1 and σ_2 differ on the y variables that correspond to these $3 \cdot |S|$ edges. Also, they differ on $|S|$ many x variables, and their $3k \cdot |S|$ copies. Therefore, overall, they differ on $(3k+1) \cdot |S| + 3 \cdot |S| = k \cdot (3k+4)$ variables. Hence, (ϕ, d) is a YES instance of Exact Differ Affine SAT.

Reverse direction. Suppose that (ϕ, d) is a YES instance of Exact Differ Affine-SAT. That is, there exist satisfying assignments σ_1 and σ_2 of ϕ that differ on $k \cdot (3k+4)$ variables. Let $S := \{v \in V(G) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } x_v\}$. We show that S is a k-sized independent set of G. Let $e(S, \bar{S})$ denote the number of edges in G that have one endpoint in S and the other endpoint in $V(G) \setminus S$. Now, let us express the number of variables on which σ_1 and σ_2 differ in terms of $|S|$ and $e(S, \bar{S})$.

Consider any edge $e = uv \in E(G)$. First, suppose that e has both its endpoints in S. Then, as σ_1 and σ_2 differ on both x_u and x_v, the expression $x_u \oplus x_v$ takes the same truth value under σ_1 and σ_2. So, as both of them satisfy the equation $x_u \oplus x_v \oplus y_e = 0$, it follows that σ_1 and σ_2 must overlap on y_e. Next, suppose that e has both its endpoints in $V(G) \setminus S$. Then, as σ_1 and σ_2 overlap on both x_u and x_v, the expression $x_u \oplus x_v$ takes the same truth value under σ_1 and σ_2. So, again, σ_1 and σ_2 must overlap on y_e. Next, suppose that e has one endpoint (say u) in S and the other endpoint in $V(G) \setminus S$. Then, as σ_1 and σ_2 differ on x_u and overlap on x_v, the expression $x_u \oplus x_v$ takes different truth values under σ_1 and σ_2. So, as both σ_1 and σ_2 satisfy the equation $x_u \oplus x_v \oplus y_e = 0$, it follows that σ_1 and σ_2 must differ on y_e. So, overall, σ_1 and σ_2 differ on $e(S, \bar{S})$ many y variables.

For any vertex $v \in V(G)$, since any satisfying assignment satisfies the equations $x_v \oplus x_v^1 = 0, x_v^1 \oplus x_v^2 = 0, \ldots, x_v^{3k-1} \oplus x_v^{3k} = 0$, it must assign the same truth value to $x_v, x_v^1, \ldots, x_v^{3k}$. So, for any $v \in S$, as σ_1 and σ_2 differ on x_v, they also differ on $x_v^1, \ldots, x_v^{3k}$. Similarly, for any $v \in V(G) \setminus S$, as σ_1 and σ_2 overlap on x_v, they also overlap on $x_v^1, \ldots x_v^{3k}$. So, overall, σ_1 and σ_2 differ on $|S|$ many x variables and their $3k \cdot |S|$ copies. Now, summing up the numbers of y variables and x variables (and their copies) on which σ_1 and σ_2 differ, we get

$$e(S, \bar{S}) + (3k+1) \cdot |S| = k \cdot (3k+4) \tag{1}$$

Let $e(S, S)$ denote the number of edges in G that have both endpoints in S. Note that

$$\sum_{v \in S} \text{degree}_G(v) = 2 \cdot e(S, S) + e(S, \bar{S})$$

Also, as G is a cubic graph, we know that $\text{degree}_G(v) = 3$ for all $v \in S$. Therefore, we get $e(S, \bar{S}) = 3 \cdot |S| - 2 \cdot e(S, S)$. Putting this expression for $e(S, \bar{S})$ in Equation (1), we have

$$(3k+4) \cdot \big(|S| - k\big) = 2 \cdot e(S, S) \tag{2}$$

If $|S| \geqslant k+1$, then LHS of Equation (1) becomes $\geqslant (3k+1) \cdot (k+1) = k \cdot (3k+4) + 1$, which is greater than its RHS. So, we must have $|S| \leqslant k$. Also, as RHS of Equation (2) is non-negative, so must be its LHS. This gives us $|S| \geqslant k$. Therefore, it follows that $|S| = k$. Putting $|S| = k$ in Equation (2), we also get $e(S, S) = 0$. That is, S is an independent set of G. Hence, (G, k) is a YES instance of INDEPENDENT SET.

This proves Theorem 1. ◀

We now turn to MAX DIFFER AFFINE-SAT, i.e, finding two solutions that differ on *at least* d variables. We show that this is hard for affine formulas of bounded arity.

▶ **Theorem 2.** *MAX DIFFER AFFINE-SAT is* NP*-hard, even on* 4*-affine formulas.*

Proof. We describe a reduction from MULTICOLORED CLIQUE ON REGULAR GRAPHS. Consider an instance (G, k) of MULTICOLORED CLIQUE, where G is a r-regular graph. We assume that each color-class of G has size $N := 2 \cdot 3^q$. It can be argued that a suitably-sized r-regular graph exists whose addition to the color-class makes this assumption hold true. We construct an affine formula ϕ as follows: For every vertex $v \in V(G)$, introduce a variable x_v and its ℓ copies $\big($say $x_v^1, x_v^2, \ldots x_v^\ell\big)$, where $\ell := k \cdot (r - k + 1) + k \cdot q$. We force these copies to take the same truth value as x_v via the equations $x_v \oplus x_v^1 = 0, x_v^1 \oplus x_v^2 = 0, \ldots, x_v^{\ell-1} \oplus x_v^\ell = 0$. For every edge $e = uv \in E(G)$, we add variables y_e and z_e, and also the equation $x_u \oplus x_v \oplus y_e \oplus z_e = 1$.

For any $1 \leqslant i \leqslant k$, consider the i^{th} color-class, say $V_i = \{v_i^1, v_i^2, \ldots, v_i^N\}$. First, we add $N/3$ many *Stage 1 dummy variables* $\big($say $d_{i,1}^1$, $d_{i,1}^2, \ldots, d_{i,1}^{N/3}\big)$, group the x variables corresponding to the vertices of V_i into $N/3$ triplets, and add $N/3$ equations that equate the xor of a triplet's variables and a dummy variable to 0. More precisely, we add the following $N/3$ equations:

$$\big(x_{v_i^1} \oplus x_{v_i^2} \oplus x_{v_i^3}\big) \oplus d_{i,1}^1 = 0,\ \big(x_{v_i^4} \oplus x_{v_i^5} \oplus x_{v_i^6}\big) \oplus d_{i,1}^2 = 0,\ \ldots,\ \big(x_{v_i^{N-2}} \oplus x_{v_i^{N-1}} \oplus x_{v_i^N}\big) \oplus d_{i,1}^{N/3} = 0$$

Next, we repeat the same process as follows: We introduce $N/3^2$ many *Stage 2 dummy variables* $\big($say $d_{i,2}^1$, $d_{i,2}^2$, $\ldots$, $d_{i,2}^{N/3^2}\big)$, group the $N/3$ many Stage 1 dummy variables into $N/3^2$ triplets, and add $N/3^2$ equations that equate the xor of a triplet's Stage 1 dummy variables and a Stage 2 dummy variable to 0. More precisely, we add the following $N/3^2$ equations:

$$\left(d_{i,1}^{1}\oplus d_{i,1}^{2}\oplus d_{i,1}^{3}\right)\oplus d_{i,2}^{1}=0,\ \left(d_{i,1}^{4}\oplus d_{i,1}^{5}\oplus d_{i,1}^{6}\right)\oplus d_{i,2}^{2}=0,\ \ldots,\ \left(d_{i,1}^{N/3-2}\oplus d_{i,1}^{N/3-1}\oplus d_{i,1}^{N/3}\right)\oplus d_{i,2}^{N/3^2}=0$$

Repeating the same procedure, we add the following $N/3^3$, $N/3^4$, ..., $N/3^q = 2$ equations:

$$\left(d_{i,2}^{1}\oplus d_{i,2}^{2}\oplus d_{i,2}^{3}\right)\oplus d_{i,3}^{1}=0,\ \left(d_{i,2}^{4}\oplus d_{i,2}^{5}\oplus d_{i,2}^{6}\right)\oplus d_{i,3}^{2}=0,\ \ldots,\ \left(d_{i,2}^{N/3^2-2}\oplus d_{i,2}^{N/3^2-1}\oplus d_{i,2}^{N/3^2}\right)\oplus d_{i,3}^{N/3^3}=0$$

$$\left(d_{i,3}^{1}\oplus d_{i,3}^{2}\oplus d_{i,3}^{3}\right)\oplus d_{i,4}^{1}=0,\ \left(d_{i,3}^{4}\oplus d_{i,3}^{5}\oplus d_{i,3}^{6}\right)\oplus d_{i,4}^{2}=0,\ \ldots,\ \left(d_{i,3}^{N/3^3-2}\oplus d_{i,3}^{N/3^3-1}\oplus d_{i,3}^{N/3^3}\right)\oplus d_{i,4}^{N/3^4}=0$$

$$\vdots$$

$$\left(d_{i,q-1}^{1}\oplus d_{i,q-1}^{2}\oplus d_{i,q-1}^{3}\right)\oplus d_{i,q}^{1}=0,\ \left(d_{i,q-1}^{4}\oplus d_{i,q-1}^{5}\oplus d_{i,q-1}^{6}\right)\oplus d_{i,q}^{2}=0$$

Next, we add $B+1$ *auxiliary variables* (say $D_i^1, \ldots, D_i^{B+1}$) and the following equations:

$$\left(d_{i,q}^{1}\oplus d_{i,q}^{2}\right)\oplus D_i^1=0,\ \left(d_{i,q}^{1}\oplus d_{i,q}^{2}\right)\oplus D_i^2=0,\ \ldots,\ \left(d_{i,q}^{1}\oplus d_{i,q}^{2}\right)\oplus D_i^{B+1}=0,$$

where $B := k\cdot(\ell+1)+k\cdot(r-k+1)+k\cdot q$ is the budget that we set on the total number of overlaps. That is, we set $d = n - B$, where n denotes the number of variables in ϕ. Now, we prove that (G, k) is a YES instance of Multicolored Clique if and only if (ϕ, d) is a YES instance of Max Differ Affine-SAT.

We first argue the forward direction. In the first assignment, we set i) all x and y variables to 0, ii) all z variables to 1, and iii) all dummy and auxiliary variables to 0. In the second assignment, we assign i) 0 to the k many x variables that correspond to the multi-colored clique's vertices, ii) 1 to the remaining x variables, iii) 0 to all z variables, iv) 0 to the $k\cdot(r-k+1)$ many y variables that correspond to those edges that have one endpoint inside the multi-colored clique and the other endpoint outside it, v) 1 to the remaining y variables, and vi) 1 to all auxiliary variables. Also, in the second assignment, for each $1 \leqslant i \leqslant k$, we assign i) 0 to that Stage 1 dummy variable which was grouped with the x variable corresponding to the multi-colored clique's vertex from the i^{th} color-class, 0 to that Stage 2 dummy variable which was grouped with this Stage 1 dummy variable, 0 to that Stage 3 dummy variable which was grouped with this Stage 2 dummy variable, and so on ..., and ii) 1 to the remaining dummy variables. It can be verified that these two assignments satisfy ϕ, and they overlap on B many variables.

We argue the reverse direction of the equivalence. First, we show that each of the k color-classes has at least one vertex on whose corresponding x variable the two assignments overlap. Consider any $1 \leqslant i \leqslant k$. Since the $B+1$ auxiliary variables are forced to take the same truth value and there are only at most B overlaps, the two assignments must differ on them. This forces the two assignments to overlap on one of the two Stage q dummy variables. Further, this forces at least one overlap amongst the three Stage $q-1$ dummy variables that were grouped with this Stage q dummy variable. This effect propagates to lower-indexed stages, and eventually forces at least one overlap amongst the x variables corresponding to the vertices of the i^{th} color-class.

Next, we show that each of the k color-classes has at most one vertex on whose corresponding x variable the two assignments overlap. Suppose not. Then, there are at least two overlaps amongst the x variables corresponding to the vertices of some color class. Also, based on the previous paragraph, we know that there is at least one overlap amongst the x variables corresponding to the vertices of each of the remaining $k-1$ color classes. Therefore, overall, there are at least $k+1$ many overlaps amongst the x variables. So, the contribution of these x variables and their copies to the total number of overlaps becomes $\geqslant (k+1)\cdot(\ell+1) = B+1$. However, this exceeds the budget B on the number of overlaps, which is a contradiction.

Based on the previous two paragraphs, we know that for each $1 \leqslant i \leqslant k$, there is exactly one overlap amongst the x variables corresponding to the vertices of the i^{th} color class. Finally, we show that the set, say S, formed by these k vertices is the desired multi-colored clique. Suppose not. Then, there are $> k \cdot (r - k + 1)$ edges that have one endpoint in S and the other endpoint outside S. Also, for each such edge, the two assignments must overlap on one of its corresponding y and z variables. Therefore, we have $> k \cdot (r - k + 1)$ overlaps on the y and z variables. Also, $k \cdot q$ overlaps are forced on the dummy variables via the equations added in the grouping procedure. Thus, overall, the total number of overlaps exceeds B, which is a contradiction. This concludes a proof sketch of Theorem 2. ◀

If, on the other hand, all equations in the formula have at most two variables, then both problems turn out to be tractable. We describe this algorithm next.

▶ **Theorem 3.** *Both* Exact Differ Affine-SAT *and* Max Differ Affine-SAT *are polynomial-time solvable on* 2*-affine formulas.*

Proof. Consider an instance (ϕ, d) of Exact Differ Affine-SAT, where ϕ is a 2-affine formula. First, we construct a graph G_0 as follows: Introduce a vertex for every variable of ϕ. For every equation of the form $x \oplus y = 0$ in ϕ, add the edge xy. We compute the connected components of G_0. Observe that for each component C of G_0, the equations of ϕ corresponding to the edges of C are simultaneously satisfied if and only if all variables of C take the same truth value. So, any pair of satisfying assignments of ϕ either overlap on all variables in C, or differ on all variables in C. Thus, we replace all variables in C by a single variable, and set its weight to be the size of C. More precisely, i) we remove all but one variable (say z) of C from the variable-set of ϕ, ii) we remove all those equations from ϕ that correspond to the edges of C, iii) for every variable $v \in C \setminus \{z\}$, we replace the remaining appearances of v in ϕ (i.e., in equations of the form $v \oplus \_ = 1$) with z, and iv) we set the weight of z to be the number of variables in C. Let ϕ' denote the variable-weighted affine formula so obtained. Then, our goal is to decide whether ϕ' has a pair of satisfying assignments such that the weights of the variables at which they differ add up to exactly d.

Note that all equations in ϕ' are of the form $x \oplus y = 1$. Next, we construct a vertex-weighted graph G_1 as follows: Introduce a vertex for every variable of ϕ', and assign it the same weight as that of its corresponding variable. For every equation $x \oplus y = 1$ of ϕ', add the edge xy. We compute the connected components of G_1. Then, we run a bipartiteness-testing algorithm on each component of G_1. Suppose that there is a component C of G_1 that is not bipartite. Then, there is an odd-length cycle in C, say with vertices $x_1, x_2, \ldots, x_{2\ell}, x_{2\ell+1}$ (in that order). Note that the edges of this cycle correspond to the equations $x_1 \oplus x_2 = 1$, $x_2 \oplus x_3 = 1, \ldots, x_{2\ell} \oplus x_{2\ell+1} = 1$, $x_{2\ell+1} \oplus x_1 = 1$ in ϕ'. Adding (modulo 2) these $2\ell + 1$ equations, we get LHS $= (2 \cdot x_1 + 2 \cdot x_2 + \ldots + 2 \cdot x_{2\ell+1}) \bmod 2 = 0$, and RHS $= (2\ell + 1) \bmod 2 = 1$. So, these $2\ell + 1$ equations of ϕ' cannot be simultaneously satisfied. Thus, we return NO. Now, assume that all components of G_1 are bipartite. See Figure 1 for an example.

Let $C_1, \ldots, C_p$ denote the connected components of G_1. Consider any $1 \leqslant i \leqslant p$. Let A and B denote the parts of the bipartite component C_i. Observe that the equations of ϕ' corresponding to the edges of C_i are simultaneously satisfied if and only if either i) all variables in A are set to 1, and all variables in B are set to 0, or ii) all variables in A are set to 0, and all variables in B are set to 1. So, any pair of satisfying assignments of ϕ' either overlap on all variables in C_i, or differ on all variables in C_i. Thus, our problem amounts to deciding whether there is a subset of components of G_1 whose collective weight is exactly d. That is, our goal is to decide whether there exists $X \subseteq [p]$ such that $\sum_{i \in X} \mathsf{weight}(C_i) = d$,

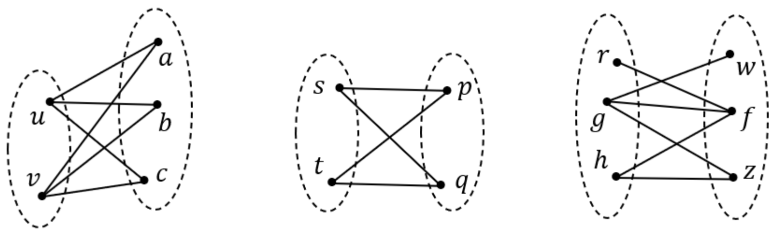

Figure 1 This figure shows the bipartite components of the graph G_1 constructed in the proof of Theorem 3, when the 2-affine formula ϕ' consists of the following equations: $u \oplus a = 1$, $u \oplus b = 1$, $u \oplus c = 1$, $v \oplus a = 1$, $v \oplus b = 1$, $v \oplus c = 1$, $s \oplus p = 1$, $s \oplus q = 1$, $t \oplus p = 1$, $t \oplus q = 1$, $r \oplus f = 1$, $g \oplus w = 1$, $g \oplus f = 1$, $g \oplus z = 1$, $h \oplus f = 1$, $h \oplus z = 1$.

where $weight(C_i)$ denotes the sum of the weights of the variables in C_i. To do so, we use the algorithm for Subset Sum problem with $\{weight(C_1), \ldots, weight(C_p)\}$ as the multi-set of integers and d as the target sum. This proves Theorem 3. The algorithm described here works almost as it is for Max Differ Affine-SAT too. In the last step, instead of reducing to Subset Sum problem, we simply check whether the collective weight of all components of G_1 is at least d. That is, if $\sum_{i=1}^{p} weight(C_i) \geqslant d$, we return YES; otherwise, we return NO. Thus, Max Differ Affine-SAT is polynomial-time solvable on 2-affine formulas. ◀

We now turn to the parameterized complexity of Exact Differ Affine-SAT and Max Differ Affine-SAT when parameterized by the number of variables that differ in the two solutions. It turns out that the exact version of the problem is W[1]-hard, while the maximization question is FPT. We first show the hardness of Exact Differ Affine-SAT by a reduction from Exact Even Set.

▶ **Theorem 4.** *Exact Differ Affine-SAT is* W[1]*-hard in the parameter* d.

Proof. We describe a reduction from Exact Even Set. Consider an instance $(\mathcal{U}, \mathcal{F}, k)$ of Exact Even Set. We construct an affine formula ϕ as follows: For every element u in the universe $\mathcal{U}$, introduce a variable x_u. For every set S in the family $\mathcal{F}$, introduce the equation $\bigoplus_{u \in S} x_u = 0$. We set $d = k$. We prove that $(\mathcal{U}, \mathcal{F}, k)$ is a YES instance of Exact Even Set if and only if (ϕ, d) is a YES instance of Exact Differ Affine-SAT. At a high level, we argue this equivalence as follows: In the forward direction, we show that the two desired satisfying assignments are i) the all 0 assignment, and ii) the assignment that assigns 1 to the variables that correspond to the elements of the given even set, and assigns 0 to the remaining variables. In the reverse direction, we show that the desired even set consists of those elements of the universe that correspond to the variables on which the two given satisfying assignments differ. We now argue the equivalence.

Forward direction. Suppose that $(\mathcal{U}, \mathcal{F}, k)$ is a YES instance of Exact Even Set. That is, there is a set $X \subseteq \mathcal{U}$ of size exactly k such that $|X \cap S|$ is even for all sets S in the family $\mathcal{F}$. Let σ_1 and σ_2 be assignments of ϕ defined as follows: For every $u \in X$, σ_1 sets x_u to 0, and σ_2 sets x_u to 1. For every $u \in \mathcal{U} \setminus X$, both σ_1 and σ_2 set x_u to 0. Note that σ_1 and σ_2 differ on exactly $|X| = k$ variables. Consider any set S in the family $\mathcal{F}$. The equation corresponding to S in the formula ϕ is $\bigoplus_{u \in S} x_u = 0$. All variables in the left-hand side are set to 0 by σ_1. Also, the number of variables in the left-hand side that are set to 1 by σ_2 is $|X \cap S|$, which is an even number. Therefore, the left-hand side evaluates to 0 under both σ_1 and σ_2. So, σ_1 and σ_2 are satisfying assignments of ϕ. Hence, (ϕ, k) is a YES instance of Exact Differ Affine-SAT.

Reverse direction. Suppose that (ϕ, k) is a YES instance of EXACT DIFFER AFFINE-SAT. That is, there are satisfying assignments σ_1 and σ_2 of ϕ that differ on exactly k variables. Let X denote the k-sized set $\{u \in \mathcal{U} \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } x_u\}$. Consider any set S in the family $\mathcal{F}$. The equation corresponding to S in the formula ϕ is $\bigoplus_{u \in S} x_u = 0$. We split the left-hand side into two parts to express this equation as $\underbrace{\bigoplus_{u \in S \setminus X} x_u}_{A} \oplus \underbrace{\bigoplus_{u \in X \cap S} x_u}_{B} = 0$. Note that σ_1 and σ_2 overlap on all variables in the first part, i.e., A. So, A evaluates to the same truth value under both assignments. Thus, as both σ_1 and σ_2 satisfy this equation, they must assign the same truth value to the second part, i.e., B, as well. Also, σ_1 and σ_2 differ on all variables in B. So, for its truth value to be same under both assignments, B must have an even number of variables. That is, $|X \cap S|$ must be even. Hence, $(\mathcal{U}, \mathcal{F}, k)$ is a YES instance of EXACT EVEN SET.

This proves Theorem 4. ◀

We now turn to the FPT algorithm for MAX DIFFER AFFINE-SAT, which is based on obtaining solutions using Gaussian elimination and working with the free variables: if the set of free variables F is "large", we can simply set them differently and force the dependent variables, and guarantee ourselves a distinction on at least $|F|$ variables. Note that this is the step that would not work as-is for the exact version of the problem. If the number of free variables is bounded, we can proceed by guessing the subset of free variables on which the two assignments differ. We make these ideas precise in the proof of Theorem 5. Also, in the proof of Theorem 6, we show that MAX DIFFER AFFINE-SAT has a polynomial kernel in the parameter d.

▶ **Theorem 5.** *MAX DIFFER AFFINE-SAT admits an algorithm with running time* $\mathcal{O}^\star(2^d)$.

Proof. Consider an instance (ϕ, d) of MAX DIFFER AFFINE-SAT. We use Gaussian elimination to find the solution set of ϕ in polynomial-time. If ϕ has no solution, we return NO. If ϕ has a unique solution and $d = 0$, we return YES. If ϕ has a unique solution and $d \geqslant 1$, we return NO. Now, assume that ϕ has multiple solutions. Let F denote the set of all free variables. Suppose that $|F| \geqslant d$. Let σ_1 denote the solution of ϕ obtained by setting all free variables to 0, and then setting the forced variables to take values as per their dependence on the free variables. Similarly, let σ_2 denote the solution of ϕ obtained by setting all free variables to 1, and then setting the forced variables to take values as per their dependence on the free variables. Note that σ_1 and σ_2 differ on all free variables (and possibly some forced variables too). So, overall, they differ on at least $|F| \geqslant d$ variables. Thus, we return YES. Now, assume that $|F| \leqslant d - 1$. We guess the subset $D \subseteq F$ of free variables on which two desired solutions (say σ_1 and σ_2) differ. Note that there are $2^{|F|} \leqslant 2^{d-1}$ such guesses.

First, consider any forced variable x that depends on an odd number of free variables from D. That is, the expression for its value is the XOR of an odd number of free variables from D (possibly along with the constant 1 and/or some free variables from $F \setminus D$). Then, note that this expression takes different truth values under σ_1 and σ_2. That is, σ_1 and σ_2 differ on x. Next, consider any forced variable x that depends on an even number of free variables from D. That is, the expression for its value is the XOR of an even number of free variables from D (possibly along with the constant 1 and/or some free variables from $F \setminus D$). Then, note that this expression takes the same truth value under σ_1 and σ_2. That is, σ_1 and σ_2 overlap on x. Thus, overall, these two solutions differ on i) all free variables from D, and ii) all those forced variables that depend upon an odd number of free variables from D. If the total count of such variables is $\geqslant d$ for some guess D, we return YES. Otherwise, we return NO. This concludes the proof. ◀

▶ **Theorem 6.** *Max Differ Affine-SAT admits a kernel with* $\mathcal{O}(d^2)$ *variables and* $\mathcal{O}(d^2)$ *equations.*

Proof. Consider an instance (ϕ, d) of Max Differ Affine-SAT. We use Gaussian elimination to find the solution set of ϕ in polynomial-time. Then, as in the proof of Theorem 5, i) we return NO if ϕ has no solution, or if ϕ has a unique solution and $d \geqslant 1$, ii) we return YES if ϕ has a unique solution and $d = 0$, or if ϕ has multiple solutions with at least d free variables. Now, assume that ϕ has multiple solutions with at most $d-1$ free variables. Note that the system of linear equations formed by the expressions for the values of forced variables is an affine formula (say ϕ') that is equivalent to ϕ. That is, ϕ' and ϕ have the same solution sets. So, we work with the instance (ϕ', d) in the remaining proof.

Suppose that there is a free variable, say x, such that at least $d-1$ forced variables depend on x. That is, there are at least $d-1$ forced variables such that the expressions for their values are the XOR of x (possibly along with the constant 1 and/or some other free variables). Let σ_1 denote the solution of ϕ' obtained by setting all free variables to 0, and then setting the forced variables to take values as per their dependence on the free variables. Let σ_2 denote the solution of ϕ' obtained by setting x to 1 and the remaining free variables to 0, and then setting the forced variables to take values as per their dependence on the free variables. Note that σ_1 and σ_2 differ on x, and also on each of the $\geqslant d-1$ forced variables that depend on x. So, overall, σ_1 and σ_2 differ on at least d variables. Thus, we return YES.

Now, assume that for every free variable x, there are at most $d-2$ forced variables that depend on x. So, as there are at most $d-1$ free variables, it follows that there are at most $(d-1)\cdot(d-2)$ forced variables that depend on at least one free variable. The remaining forced variables are the ones that do not depend on any free variable. That is, any such forced variable y is set to a constant (i.e., 0 or 1) as per the expression for its value. We remove the variable y and its corresponding equation (i.e., $y = 0$ or $y = 1$) from ϕ', and we leave d unchanged. This is safe because y takes the same truth value under all solutions of ϕ'. Note that the affine formula so obtained has at most $d-1$ free variables and at most $(d-1)\cdot(d-2)$ forced variables. So, overall, it has at most $(d-1)^2$ variables. Also, it has at most $(d-1)\cdot(d-2)$ equations. This concludes the proof. ◀

We finally turn to the "dual" parameter, $n-d$: the number of variables on which the two assignments sought *overlap*. We show that both the exact and maximization variants for affine formulas are W[1]-hard in this parameter by reductions from Exact Odd Set and Odd Set, respectively.

▶ **Theorem 7.** *The problems Exact Differ Affine-SAT and Max Differ Affine-SAT are* W[1]*-hard in the parameter* $n-d$.

Proof. We describe a reduction from Exact Odd Set. Consider an instance $(\mathcal{U}, \mathcal{F}, k)$ of Exact Odd Set. We construct an affine formula ϕ as follows: For every element u in the universe $\mathcal{U}$, introduce a variable x_u. For every odd-sized set S in the family $\mathcal{F}$, introduce the equation $\bigoplus_{u \in S} x_u = 1$. For every even-sized set S in the family $\mathcal{F}$, introduce $k+1$ variables $y_S, z_S^1, z_S^2, \ldots, z_S^k$, and the equations $y_S \oplus z_S^1 = 0, y_S \oplus z_S^2 = 0, \ldots, y_S \oplus z_S^k = 0$ and $\bigoplus_{u \in S} x_u \oplus y_S = 0$. The number of variables in ϕ is $n = |\mathcal{U}| + (k+1) \cdot \big|\{S \in \mathcal{F} \mid |S| \text{ is even}\}\big|$. We set $d = n-k$. We prove that $(\mathcal{U}, \mathcal{F}, k)$ is a YES instance of Exact Odd Set if and only if (ϕ, d) is a YES instance of Exact Differ Affine-SAT. At a high level, we argue this equivalence as follows: In the forward direction, we show that the two desired satisfying assignments are i) the assignment that sets all y and z variables to 0 and all x variables to 1, and ii) the assignment that sets all y and z variables to 1, assigns 1 to all those x variables that correspond to the elements of the given odd set, and assigns 0 to the remaining x variables. In the reverse direction, we show that the two assignments must differ on all y and z variables (and so, all k overlaps are restricted to occur at x variables), and the desired odd set consists of those elements of the universe that correspond to the x variables on which the two assignments overlap. We present a full proof of this equivalence in Theorem 7. This reduction also works with Odd Set as the source problem and Max Differ Affine-SAT as the target problem. So, Max Differ Affine-SAT is also W[1]-hard in the parameter $n - d$.

Forward direction. Suppose that $(\mathcal{U}, \mathcal{F}, k)$ is a YES instance of Exact Differ Affine-SAT. That is, there is a set $X \subseteq \mathcal{U}$ of size exactly k such that $|X \cap S|$ is odd for all sets S in the family $\mathcal{F}$. Let σ_1 and σ_2 be assignments of ϕ defined as follows: For every even-sized set S in the family $\mathcal{F}$, σ_1 sets $y_S, z_S^1, z_S^2, \ldots, z_S^k$ to 0, and σ_2 sets $y_S, z_S^1, z_S^2, \ldots, z_S^k$ to 1. For every $u \in X$, both σ_1 and σ_2 set x_u to 1. For every $u \in \mathcal{U} \setminus X$, σ_1 sets x_u to 1, and σ_2 sets x_u to 0. Note that σ_1 and σ_2 overlap on exactly $|X| = k$ variables (and so, they differ on exactly $n - k$ variables). Now, we show that σ_1 and σ_2 are satisfying assignments of ϕ.

First, we argue that σ_1 and σ_2 satisfy the equations of ϕ that were added corresponding to odd-sized sets of the family $\mathcal{F}$. Consider any odd-sized set S in the family $\mathcal{F}$. The equation corresponding to S in the formula ϕ is $\bigoplus_{u \in S} x_u = 1$. The number of variables in the left-hand side that are set to 1 by σ_2 is $|X \cap S|$, which is an odd number. Also, all $|S|$ (again, which is an odd number) variables in the left-hand side are set to 1 by σ_1. Therefore, the left-hand side evaluates to 1 under both σ_1 and σ_2. So, both these assignments satisfy the equation $\bigoplus_{u \in S} x_u = 1$.

Next, we argue that σ_1 and σ_2 satisfy the equations of ϕ that were added corresponding to even-sized sets of the family $\mathcal{F}$. Consider any even-sized set S in the family $\mathcal{F}$. The $k+1$ equations corresponding to S in the formula ϕ are $y_S \oplus z_S^1 = 0, y_S \oplus z_S^2 = 0, \ldots, y_S \oplus z_S^k = 0$ and $\bigoplus_{u \in S} x_u \oplus y_S = 0$. Consider any of the first k equations, say $y_S \oplus z_S^i = 0$, where $1 \leqslant i \leqslant k$. Both variables on the left-hand side, i.e., y_S and z_S^i, are assigned the same truth value, i.e., both 0 by σ_1 and both 1 by σ_2. So, both these assignments satisfy the equation $y_S \oplus z_S^i = 0$. Next, consider the last equation, i.e., $\bigoplus_{u \in S} x_u \oplus y_S = 0$. The number of variables amongst $x_u|_{u \in S}$ that are set to 1 by σ_2 is $|X \cap S|$, which is an odd number. Also, the variable y_S is set to 1 by σ_2. Therefore, overall, the number of variables in the left-hand side that are set to 1 by σ_2 is even. Also, σ_1 sets all variables on the left-hand side to 1 except y_S. That is, it sets all the $|S|$ (again, which is an even number) variables $x_u|_{u \in S}$ to 1. Therefore, the left-hand side evaluates to 0 under both σ_1 and σ_2. So, both these assignments satisfy the equation $\bigoplus_{u \in S} x_u \oplus y_S = 0$.

Hence, $(\phi, n - k)$ is a YES instance of Exact Differ Affine-SAT.

Reverse direction. Suppose that $(\phi, n - k)$ is a YES instance of Exact Differ Affine-SAT. That is, there are satisfying assignments σ_1 and σ_2 of ϕ that overlap on exactly k variables. Consider any even-sized set S in the family $\mathcal{F}$. As σ_1 satisfies the equations $y_S \oplus z_S^1 = 0, y_S \oplus z_S^2 = 0, \ldots, y_S \oplus z_S^k = 0$, it must assign the same truth value to all the $k+1$ variables $y_S, z_S^1, z_S^2, \ldots, z_S^k$. Similarly, σ_2 must assign the same truth value to $y_S, z_S^1, z_S^2, \ldots, z_S^k$. Therefore, either σ_1 and σ_2 overlap on all these $k+1$ variables, or they differ on all these $k+1$ variables. So, as there are only k overlaps, σ_1 and σ_2 must differ on $y_S, z_S^1, z_S^2, \ldots, z_S^k$. Thus, all the k overlaps occur at x variables. Let X denote the k-sized set $\{u \in \mathcal{U} \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } x_u\}$. Now, we show that $|X \cap S|$ is odd for all sets S in $\mathcal{F}$.

First, we argue that X has odd-sized intersection with all odd-sized sets of the family $\mathcal{F}$. Consider any odd-sized set S in the family $\mathcal{F}$. The equation corresponding to S in the formula ϕ is $\bigoplus_{u \in S} x_u = 1$. We split the left-hand side into two parts to express this equation as $\underbrace{\bigoplus_{u \in X \cap S} x_u}_{A} \oplus \underbrace{\bigoplus_{u \in S \setminus X} x_u}_{B} = 1$. Note that σ_1 and σ_2 overlap on all variables in the first part, i.e., A. So, A evaluates to the same truth value under both assignments. Thus, as both σ_1 and σ_2 satisfy this equation, they must assign the same truth value to the second part, i.e., B, as well. Also, σ_1 and σ_2 differ on all variables in B. So, for its truth value to be same under both assignments, B must have an even number of variables. That is, $|S \setminus X|$ must be even. Now, as $|S|$ is odd and $|S \setminus X|$ is even, we infer that $|X \cap S| = |S| - |S \setminus X|$ is odd.

Next, we argue that X has odd-sized intersection with all even-sized sets of the family $\mathcal{F}$. Consider any even-sized set S in the family $\mathcal{F}$. Amongst the $k+1$ equations corresponding to S in the formula ϕ, consider the last equation, i.e., $\bigoplus_{u \in S} x_u \oplus y_S = 0$. We split the left-hand side into two parts to express this equation as $\underbrace{\bigoplus_{u \in X \cap S} x_u}_{A} \oplus \underbrace{\bigoplus_{u \in S \setminus X} x_u \oplus y_S}_{B} = 0$. Note that σ_1 and σ_2 overlap on all variables in the first part, i.e., A. So, A evaluates to the same truth value under both assignments. Thus, as both σ_1 and σ_2 satisfy this equation, they must assign the same truth value to the second part, i.e., B. Also, σ_1 and σ_2 differ on all variables in B. So, for its truth value to be same under both assignments, B must have an even number of variables. That is, $|S \setminus X| + 1$ must be even. Now, as $|S|$ is even and $|S \setminus X|$ is odd, we infer that $|X \cap S| = |S| - |S \setminus X|$ is odd. Hence, $(\mathcal{U}, \mathcal{F}, k)$ is a YES instance of Exact Odd Set. ◀

4 2-CNF formulas

In this section, we explore the classical and parameterized complexity of Max Differ 2-SAT and Exact Differ 2-SAT. We first show that these problems are polynomial time solvable on $(2, 2)$-CNF formulas by constructing a graph corresponding to the instance and observing some structural properties of that graph. Then we show that both of these problems are W[1]-hard with respect to the parameter d. We begin by proving the following theorem.

▶ **Theorem 8.** *Max Differ 2-SAT is polynomial-time solvable on $(2, 2)$-CNF formulas.*

We use similar ideas in the proof of Theorem 8 to show that Exact Differ 2-SAT can also be solved in polynomial time on $(2, 2)$-CNF formulas. This requires more careful analysis of the graph constructed and a reduction to Subset Sum problem, as we want the individual contributions, in terms of number of variables where the assignments differ, to sum up to an exact value. We show the result in the following theorem.

▶ **Theorem 9.** *Exact Differ 2-SAT is polynomial-time solvable on $(2, 2)$-CNF formulas.*

Due to lack of space, the proofs of these results are deferred to a full version of the paper. Looking at the parameterized complexity of Exact Differ 2-SAT and Max Differ 2-SAT with respect to the parameter d, we establish the following hardness result.

▶ **Theorem 10.** *Exact/Max Differ 2-SAT is W[1]-hard in the parameter d.*

We describe a reduction from Independent Set. Consider an instance (G, k) of Independent Set. We construct a 2-CNF formula ϕ as follows: For every vertex $v \in V(G)$, introduce two variables x_v and y_v; we refer to them as x-variable and y-variable respectively. For every edge $uv \in E(G)$, i) we add a clause that consists of the x-variables corresponding to the vertices u and v, i.e., $x_u \vee x_v$, and ii) we add a clause that consists of the y-variables corresponding to the vertices u and v, i.e., $y_u \vee y_v$. For every pair of vertices $u, v \in V(G)$, we add a clause that consists of the x-variable corresponding to u and the y-variable corresponding to v, i.e., $x_u \vee y_v$. We set $d = 2k$. We prove that (G, k) is a YES instance of Independent Set if and only if (ϕ, d) is a YES instance of Exact Differ 2-SAT.

At a high level, we argue this equivalence as follows: In the forward direction, we show that the two desired satisfying assignments are i) the assignment that assigns 0 to all x-variables corresponding to the vertices of the given independent set, and 1 to the remaining variables, and ii) the assignment that assigns 0 to all y-variables corresponding to the vertices of the given independent set, and 1 to the remaining variables. In the reverse direction, we partition the set of variables on which the two given assignments differ into two parts: i) one part consists of those variables that are set to 1 by the first assignment, and 0 by the second assignment, and ii) the other part consists of those variables that are set to 0 by the first assignment, and 1 by the second assignment. Then, we show that at least one of these two parts has the desired size, and it is not a mix of x-variables and y-variables. That is, either it has only x-variables, or it has only y-variables. Finally, we show that the vertices that correspond to the variables in this part form the desired independent set. A detailed proof of equivalence is deferred to a full version of the paper.

5 Hitting formulas

In this section, we consider hitting formulas, and we show that both its diverse variants, i.e., Exact Differ Hitting-SAT and Max Differ Hitting-SAT, are polynomial-time solvable.

▶ **Theorem 11.** *Exact Differ Hitting-SAT admits a polynomial-time algorithm.*

Proof. Consider an instance (ϕ, d) of Exact Differ Hitting-SAT, where ϕ is a hitting formula with m clauses (say $C_1, \ldots, C_m$) on n variables. For every $1 \leqslant i \leqslant m$, let $vars(C_i)$ denote the set of all variables that appear in the clause C_i. For every $1 \leqslant i, j \leqslant m$, let $\lambda(i, j)$ denote the number of variables $x \in vars(C_i) \cap vars(C_j)$ such that x appears as a positive literal in one clause, and as a negative literal in the other clause. Note that

$$\begin{aligned}
&\big|\{(\sigma_1, \sigma_2) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } d \text{ variables, and both } \sigma_1 \text{ and } \sigma_2 \text{ satisfy } \phi\}\big| \\
&= \big|\{(\sigma_1, \sigma_2) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } d \text{ variables}\}\big| \\
&\quad - \big|\{(\sigma_1, \sigma_2) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } d \text{ variables, and } \sigma_1 \text{ falsifies } \phi\}\big| \\
&\quad - \big|\{(\sigma_1, \sigma_2) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } d \text{ variables, and } \sigma_2 \text{ falsifies } \phi\}\big| \\
&\quad + \big|\{(\sigma_1, \sigma_2) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } d \text{ variables, and both } \sigma_1 \text{ and } \sigma_2 \text{ falsify } \phi\}\big| \\
&= 2^n \cdot \binom{n}{d} - \big|\{\sigma_1 \mid \sigma_1 \text{ falsifies } \phi\}\big| \cdot \binom{n}{d} - \big|\{\sigma_2 \mid \sigma_2 \text{ falsifies } \phi\}\big| \cdot \binom{n}{d} \\
&\quad + \sum_{i=1}^{m} \sum_{j=1}^{m} \underbrace{\big|\{(\sigma_1, \sigma_2) \mid \sigma_1 \text{ and } \sigma_2 \text{ differ on } d \text{ variables, } \sigma_1 \text{ falsifies } C_i \text{, and } \sigma_2 \text{ falsifies } C_j\}\big|}_{\alpha(i,j)} \\
&= \left(2^n - 2 \cdot \sum_{i=1}^{m} 2^{n - |vars(C_i)|}\right) \cdot \binom{n}{d} + \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha(i, j)
\end{aligned}$$

Consider any $1 \leqslant i, j \leqslant m$. Let us find an expression for $\alpha(i, j)$. That is, let us count the number of pairs (σ_1, σ_2) of assignments of ϕ such that σ_1 and σ_2 differ on d variables, σ_1 falsifies C_i, and σ_2 falsifies C_j. Since σ_1 falsifies C_i, it must set every variable in $vars(C_i)$

such that its corresponding literal in the clause C_i is falsified. That is, for every $x \in vars(C_i)$, if x appears as a positive literal in C_i, then σ_1 must set x to 0; otherwise, it must set x to 1. Similarly, since σ_2 falsifies C_j, it must set every variable in $vars(C_j)$ such that its corresponding literal in the clause C_j is falsified.

There is just one choice for the truth values assigned to the variables in $vars(C_i) \cap vars(C_j)$ by σ_1 and σ_2. Also, note that for every variable x in $vars(C_i) \cap vars(C_j)$, if x appears as a positive literal in one clause and as a negative literal in the other clause, then σ_1 and σ_2 differ on x; otherwise, they overlap on x. So, overall, σ_1 and σ_2 differ on $\lambda(i,j)$ variables amongst the variables in $vars(C_i) \cap vars(C_j)$.

We go over all possible choices for the numbers of variables on which σ_1 and σ_2 differ (say d_1, d_2 and d_3 many variables) amongst the variables in $vars(C_i) \setminus vars(C_j), vars(C_j) \setminus vars(C_i)$ and $vars(\phi) \setminus (vars(C_i) \cup vars(C_j))$ respectively, where $vars(\phi)$ denotes the set of all variables of ϕ. As σ_1 and σ_2 differ on d variables in total, we have $\lambda(i,j)+d_1+d_2+d_3 = d$.

There is just one choice for the truth values assigned to the variables in $vars(C_i) \setminus vars(C_j)$ by σ_1, and there are $\binom{|vars(C_i) \setminus vars(C_j)|}{d_1}$ choices for the truth values assigned to the variables in $vars(C_i) \setminus vars(C_j)$ by σ_2. Similarly, there is just one choice for the truth values assigned to the variables in $vars(C_j) \setminus vars(C_i)$ by σ_2, and there are $\binom{|vars(C_j) \setminus vars(C_i)|}{d_2}$ choices for the truth values assigned to the variables in $vars(C_j) \setminus vars(C_i)$ by σ_1.

There are $\binom{n-|vars(C_i) \cup vars(C_j)|}{d_3}$ choices for the d_3 variables on which σ_1 and σ_2 differ amongst the variables in $vars(\phi) \setminus (vars(C_i) \cup vars(C_j))$. For each variable x amongst these d_3 variables, there are two ways in which σ_1 and σ_2 can assign truth values to x. That is, either i) σ_1 sets x to 0 and σ_2 sets x to 1, or ii) σ_1 sets x to 1 and σ_2 sets x to 0. For each variable x amongst the remaining $n - |vars(C_i) \cup vars(C_j)| - d_3$ variables, there are again two ways in which σ_1 and σ_2 can assign truth values to x. That is, either i) both σ_1 and σ_2 set x to 1, or ii) both σ_1 and σ_2 set x to 0. So, overall, the number of ways in which σ_1 and σ_2 can assign truth values to the variables in $vars(\phi) \setminus (vars(C_i) \cup vars(C_j))$ is $\binom{n-|vars(C_i) \cup vars(C_j)|}{d_3} \cdot 2^{d_3} \cdot 2^{n-|vars(C_i) \cup vars(C_j)|-d_3}$.

Thus, we get the following expression for $\alpha(i,j)$:

$$2^{n-|vars(C_i) \cup vars(C_j)|} \cdot \sum_{\substack{d_1,d_2,d_3 \geqslant 0: \\ d_1+d_2+d_3=d-\lambda(i,j)}} \binom{|vars(C_i) \setminus vars(C_j)|}{d_1} \binom{|vars(C_j) \setminus vars(C_i)|}{d_2} \binom{n-|vars(C_i) \cup vars(C_j)|}{d_3}$$

Plugging this into the previously obtained equality, we get an expression to count the number of pairs (σ_1, σ_2) of satisfying assignments of ϕ that differ on d variables. This expression can be evaluated in polynomial-time. If the count so obtained is non-zero, we return YES; otherwise, we return NO. This proves Theorem 11. Note that (ϕ, d) is a YES instance of Max Differ Hitting-SAT if and only if at least one of $(\phi, d), (\phi, d+1), \ldots, (\phi, n)$ is a YES instance of Exact Differ Hitting-SAT. Thus, as Exact Differ Hitting-SAT is polynomial-time solvable, so is Max Differ Hitting-SAT. ◀

6 Concluding remarks

In this work, we undertook a complexity-theoretic study of the problem of finding a diverse pair of satisfying assignments of a given Boolean formula, when restricted to affine, 2-CNF and hitting formulas. This problem can also be studied for i) other classes of formulas on which SAT is polynomial-time solvable, ii) more than two solutions, and iii) other notions of distance between assignments. An immediate open problem is to resolve the parameterized complexities of Exact Differ 2-SAT and Max Differ 2-SAT in the parameter $n - d$.

References

1 Ola Angelsmark and Johan Thapper. Algorithms for the maximum Hamming distance problem. In *International Workshop on Constraint Solving and Constraint Logic Programming*, pages 128–141. Springer, 2004. doi:10.1007/11402763_10.

2 Bengt Aspvall, Michael F Plass, and Robert Endre Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information processing letters*, 8(3):121–123, 1979. doi:10.1016/0020-0190(79)90002-4.

3 Julien Baste, Michael R Fellows, Lars Jaffke, Tomáš Masařík, Mateus de Oliveira Oliveira, Geevarghese Philip, and Frances A Rosamond. Diversity of solutions: An exploration through the lens of fixed-parameter tractability theory. *Artificial Intelligence*, 303:103644, 2022. doi:10.1016/j.artint.2021.103644.

4 Julien Baste, Lars Jaffke, Tomáš Masařík, Geevarghese Philip, and Günter Rote. FPT algorithms for diverse collections of hitting sets. *Algorithms*, 12(12):254, 2019. doi:10.3390/a12120254.

5 Endre Boros, Yves Crama, and Peter L Hammer. Polynomial-time inference of all valid implications for horn and related formulae. *Annals of Mathematics and Artificial Intelligence*, 1:21–32, 1990. doi:10.1007/BF01531068.

6 Endre Boros, Peter L Hammer, and Xiaorong Sun. Recognition of q-Horn formulae in linear time. *Discrete Applied Mathematics*, 55(1):1–13, 1994. doi:10.1016/0166-218X(94)90033-7.

7 Michele Conforti, Gérard Cornuéjols, Ajai Kapoor, Kristina Vuškovic, and MR Rao. Balanced matrices. *Mathematical Programming: State of the Art*, pages 1–33, 1994.

8 Stephen A Cook. The complexity of theorem-proving procedures. In *Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook*, pages 143–152. Association for Computing Machinery, 2023. doi:10.1145/3588287.3588297.

9 Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*, volume 5. Springer, 2015. doi:10.1007/978-3-319-21275-3.

10 Vilhelm Dahllöf. Algorithms for max Hamming exact satisfiability. In *Proceedings of the 16th International Symposium on Algorithms and Computation (ISAAC)*, pages 829–838. Springer, 2005. doi:10.1007/11602613_83.

11 Mark de Berg, Andrés López Martínez, and Frits C. R. Spieksma. Finding diverse minimum s-t cuts. In *Proceedings of the 34th International Symposium on Algorithms and Computation ISAAC*, volume 283 of *LIPIcs*, pages 24:1–24:17, 2023. doi:10.4230/LIPIcs.ISAAC.2023.24.

12 William F Dowling and Jean H Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *The Journal of Logic Programming*, 1(3):267–284, 1984. doi:10.1016/0743-1066(84)90014-1.

13 Rod G Downey, Michael R Fellows, Alexander Vardy, and Geoff Whittle. The parametrized complexity of some fundamental problems in coding theory. *SIAM Journal on Computing*, 29(2):545–570, 1999. doi:10.1137/S0097539797323571.

14 Rodney G Downey and Michael R Fellows. Fixed-parameter intractability. In *1992 Seventh Annual Structure in Complexity Theory Conference*, pages 36–37. IEEE Computer Society, 1992.

15 Rodney G Downey, Michael R Fellows, et al. *Fundamentals of Parameterized Complexity*, volume 4. Springer, 2013.

16 S. Even, A. Shamir, and A. Itai. On the complexity of time table and multi-commodity flow problems. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science (SFCS)*, pages 184–193. IEEE Computer Society, 1975. doi:10.1109/SFCS.1975.21.

17 Fedor V Fomin, Petr A Golovach, Lars Jaffke, Geevarghese Philip, and Danil Sagunov. Diverse pairs of matchings. In *31st International Symposium on Algorithms and Computation (ISAAC 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

18 Fedor V Fomin, Petr A Golovach, Fahad Panolan, Geevarghese Philip, and Saket Saurabh. Diverse collections in matroids and graphs. *Mathematical Programming*, pages 1–33, 2023.

19 John Franco and Allen Van Gelder. A perspective on certain polynomial-time solvable classes of satisfiability. *Discrete Applied Mathematics*, 125(2-3):177–214, 2003. doi:10.1016/S0166-218X(01)00358-4.

20 Aadityan Ganesh, HV Vishwa Prakash, Prajakta Nimbhorkar, and Geevarghese Philip. Disjoint stable matchings in linear time. In *Proceedings of the 47th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 94–105. Springer, 2021. doi:10.1007/978-3-030-86838-3_7.

21 Joseph F Grcar. How ordinary elimination became Gaussian elimination. *Historia Mathematica*, 38(2):163–218, 2011.

22 Tesshu Hanaka, Yasuaki Kobayashi, Kazuhiro Kurita, and Yota Otachi. Finding diverse trees, paths, and more. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3778–3786, 2021. doi:10.1609/aaai.v35i5.16495.

23 Gordon Hoi, Sanjay Jain, and Frank Stephan. A fast exponential time algorithm for max hamming distance x3sat. *arXiv preprint*, 2019. arXiv:1910.01293.

24 Gordon Hoi and Frank Stephan. Measure and conquer for max Hamming distance XSAT. In *30th International Symposium on Algorithms and Computation (ISAAC 2019)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

25 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *Journal of Computer and System Sciences (JCSS)*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

26 Kazuo Iwama. CNF-satisfiability test by counting and polynomial average time. *SIAM Journal on Computing*, 18(2):385–391, 1989. doi:10.1137/0218026.

27 Richard M Karp. *Reducibility among combinatorial problems*. Springer, 2010. doi:10.1007/978-3-540-68279-0_8.

28 Konstantinos Koiliaris and Chao Xu. Faster pseudopolynomial time algorithms for subset sum. *ACM Transactions on Algorithms (TALG)*, 15(3):1–20, 2019. doi:10.1145/3329863.

29 Melven R Krom. The decision problem for a class of first-order formulas in which all disjunctions are binary. *Mathematical Logic Quarterly*, 13(1-2):15–20, 1967.

30 Leonid Anatolevich Levin. Universal sequential search problems. *Problemy peredachi informatsii*, 9(3):115–116, 1973.

31 Harry R Lewis. Renaming a set of clauses as a Horn set. *Journal of the ACM (JACM)*, 25(1):134–135, 1978. doi:10.1145/322047.322059.

32 Ilya Mironov and Lintao Zhang. Applications of sat solvers to cryptanalysis of hash functions. In *Proceedings of the 9th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, pages 102–115. Springer, 2006. doi:10.1007/11814948_13.

33 Neeldhara Misra, Harshil Mittal, and Saraswati Nanoti. Diverse non crossing matchings. In *CCCG*, pages 249–256, 2022.

34 Bojan Mohar. Face covers and the genus problem for apex graphs. *Journal of Combinatorial Theory (JCT), Series B*, 82(1):102–117, 2001. doi:10.1006/jctb.2000.2026.

35 Thomas J Schaefer. The complexity of satisfiability problems. In *Proceedings of the tenth annual ACM Symposium on Theory of Computing*, pages 216–226, 1978. doi:10.1145/800133.804350.

36 Maria Grazia Scutella. A note on Dowling and Gallier's top-down algorithm for propositional horn satisfiability. *The Journal of Logic Programming*, 8(3):265–273, 1990. doi:10.1016/0743-1066(90)90026-2.

37 Yakir Vizel, Georg Weissenbacher, and Sharad Malik. Boolean satisfiability solvers and their applications in model checking. *Proceedings of the IEEE*, 103(11):2021–2035, 2015. doi:10.1109/JPROC.2015.2455034.