A CNF Class Generalizing Exact Linear Formulas

Stefan Porschen and Ewald Speckenmeyer

Institut für Informatik, Universität zu Köln,
Pohligstr. 1, D-50969 Köln, Germany
{porschen,esp}@informatik.uni-koeln.de

Abstract. The *fibre view* on clause sets, previously introduced in [12], is used in the present paper to define and investigate subclasses of CNF that appear to be polynomial time solvable w.r.t. SAT. The most interesting of these classes is a generalization of exact linear formulas, namely formulas such that each pair of distinct clauses has all variables in common or exactly one. By definition, in an exact linear formula each pair of distinct clauses has exactly one variable in common. SAT-solving for exact linear formulas was shown to be easy in [14]. Here we provide an algorithm solving the decision and counting variants of SAT for the generalized class in polynomial time. Moreover we study some other structurally defined formula classes on the basis of the fibre view. We show that these classes have the property that their members all are satisfiable or all are unsatisfiable.

Keywords: CNF satisfiability, exact linear formula, hypergraph, fibre-transversal.

1 Introduction

Exploiting the *fibre view* on clause sets recently introduced in [12] we consider some structurally defined subclasses of CNF regarding their behaviour w.r.t. SAT. The most interesting of these classes yields a generalization of *exact linear formulas*. The class of linear CNF formulas has recently been studied in [14] revealing its general NP-completeness w.r.t. SAT. Each pair of distinct clauses of a linear formula by definition has at most one variable in common. And requiring that there should be *exactly one* variable in the intersection of the variable sets of each pair of distinct clauses, one arrives at the subclass of exact linear formulas. SAT-solving was shown to be easy for exact linear formulas in [14]. Here we consider the class of formulas where each pair of distinct clauses has all variables in common or exactly one, thus extending exact linear formulas. The members of this class are called *exact linearly-based* formulas. We design an algorithm providing polynomial-time SAT-decidability for this class. Moreover, by a slight modification we are also able to show that the counting variant #-SAT for exact linearly-based formulas belongs to P. Furthermore, we study some other structurally defined formula classes using the fibre view on clause sets, and show that

H. Kleine Büning and X. Zhao (Eds.): SAT 2008, LNCS 4996, pp. 231–245, 2008.

they behave trivially w.r.t. SAT meaning that its members always are satisfiable. There are known several classes, for which SAT can be tested in polynomial time, such as quadratic formulas, (extended and q-)Horn formulas, matching formulas etc. [1,3,4,5,6,8,9,10,16,17]. The classes studied in this paper appear not to belong to one of these classes. On the other hand, mixing polynomial-time classes, in general, yields classes for which SAT becomes NP-complete, as already is the case for Horn and quadratic formulas [13], cf. also [8]. Closely related to some of these classes is the classic theorem of Schaefer in [15]. That theorem classifies satisfiability problems w.r.t. their complexity. The theorem does not automatically apply if restrictions on the number of occurrences of variables in CNF formulas are valid explicitly or implicitly. E.g. in [7] it is shown that whereas unrestricted k-SAT is NP-complete, for $k \geq 3$, it behaves trivially (i.e. all formulas are satisfiable) if each clause has length exactly k and no variable occurs in more than $f(k)$ clauses; it gets NP-complete then if variables are allowed to occur at most $f(k)+1$ times. Here $f(k)$ asymptotically grows as $\lfloor 2^k/(e \cdot k) \rfloor$; this bound has meanwhile been improved by other authors. However, it seems to be unknown whether one can expect a dichotomy result like Schaefer's regarding the occurrence number. The classes studied in the present paper do not meet the requirement that all clauses have a constant equal length, but seem to have a hidden or implicit bound on the maximal number of occurences along with additional structure.

The paper is organized as follows: The next section collects the notation and some preliminaries. Section 3 describes the fibre view on clause sets, and elaborates a subclass only containing unsatisfiable members. Section 4 contains the main part of the paper, namely showing that the decision, search and counting variants of SAT for exact linearly-based formulas can be solved in polynomial time. In Section 5 we apply the fibre view concept on some further CNF classes showing that they behave trivially w.r.t. SAT. Finally, in Section 6 we collect some conclusions and open problems.

2 Notation and Preliminaries

To fix the notation, let CNF denote the set of duplicate-free conjunctive normal form formulas over propositional variables $x \in \{0,1\}$. A *positive (negative)* literal is a (negated) variable. The *negation (complement)* of a literal l is $\bar{l}$. Each formula $C \in$ CNF is considered as a clause set, and each clause $c \in C$ is represented as a literal set which in addition is assumed to be free of complemented pairs $\{x, \bar{x}\}$. For formula C, clause c, literal l, by $V(C), V(c), V(l)$ we denote the variables contained (neglecting negations), correspondingly. $L(C)$ is the set of all literals in C. The length of C is denoted as $\|C\|$. For $U \subset L(C)$, let $C(U) := \{c \in C | c \cap U \neq \varnothing\}$; we simply write $C(l)$, if $U = \{l\}$. The satisfiability problem (SAT) asks, whether input $C \in$ CNF has a *model*, which is a truth assignment $t : V(C) \to \{0,1\}$ assigning at least one literal in each clause of C to 1. For $C \in$ CNF, let $M(C)$ be the space of all models of C and let UNSAT := CNF − SAT. It is convenient to identify truth assignments with $|V|$-clauses in the following

simple way: Let $x^0 := \bar{x}$, $x^1 := x$. Then we can identify $t : V \to \{0,1\}$ with the literal set $\{x^{t(x)} | x \in V\}$, and, for $b \subset V$, the *restriction* $t|_b$ is identified with the literal set $\{x^{t(x)} | x \in b\}$. The collection of all literal sets obtained as just described by running through all total truth assignments $V \to \{0,1\}$ is denoted as W_V. We call W_V the *hypercube (hc) formula (over V)*, since its clauses correspond 1:1 to the vertices of a hypercube of dimension $|V|$. E.g., for $V = \{x, y\}$, we have $W_V = \{xy, \bar{x}y, x\bar{y}, \bar{x}\bar{y}\}$ writing clauses as literal strings. For a clause c, we denote by c^γ the clause in which all its literals are complemented. Similarly, let $t^\gamma = 1 - t : V \to \{0,1\} \in W_V$, $C^\gamma := \{c^\gamma | c \in C\}$, and for $\mathcal{C} \subseteq$ CNF, let $\mathcal{C}^\gamma := \{C^\gamma | C \in \mathcal{C}\}$. We call C *asymmetric* if for each $c \in C$ we have $c^\gamma \notin C$. Asym $\subset$ CNF, denotes the set of all asymmetric formulas.

3 The Fibre View on Clause Sets

The *fibre view*, as introduced in [12], regards a clause set C as composed of *fibres* over a hypergraph: All clauses c of C *projecting* onto the same variable set $b = V(c)$, when negations are eliminated, form the *fibre* C_b *over* b, namely $C_b = \{c \in C | V(c) = b\}$. The collection of these *base elements* b forms a hypergraph, the *base hypergraph* $\mathcal{H}(C) = (V(C), B(C))$ of C, where $B(C) = \{b := V(c) | c \in C\}$. Hence, C is the disjoint union of all its fibres: $C = \bigcup_{b \in B(C)} C_b$. Conversely, we can also start with a given arbitrary hypergraph $\mathcal{H} = (V, B)$ serving as a base hypergraph if its vertices $x \in V$ are regarded as Boolean variables such that for each $x \in V$ there is a (hyper)edge $b \in B$ containing x. Recall that, for any $b \in B$, W_b is the hypercube formula over the set of variables in b. Then the set of all clauses over $\mathcal{H}$ is $K_\mathcal{H} := \bigcup_{b \in B} W_b$, also called the *total clause set over* $\mathcal{H}$. W_b is the fibre of $K_\mathcal{H}$ over b. For example, given $V = \{x_1, x_2, x_3\}$, and $B = \{b_1 := x_1, b_2 := x_1x_2, b_3 := x_1x_3\}$, we have $K_\mathcal{H} = W_{b_1} \cup W_{b_2} \cup W_{b_3}$, where $W_{b_1} = \{x_1,\ \bar{x}_1\}$, $W_{b_2} = \{x_1x_2,\ \bar{x}_1x_2,\ x_1\bar{x}_2,\ \bar{x}_1\bar{x}_2\}$, and $W_{b_3} = \{x_1x_3,\ \bar{x}_1x_3,\ x_1\bar{x}_3,\ \bar{x}_1\bar{x}_3\}$ are the hc formulas over b_1, b_2, and b_3, respectively.

A *formula over* $\mathcal{H}$ (or a *$\mathcal{H}$-based)* formula is a subset $C \subseteq K_\mathcal{H}$ such that $C_b := C \cap W_b \neq \varnothing$, for each $b \in B$. Given a $\mathcal{H}$-based formula $C \subseteq K_\mathcal{H}$ with the additional property that $\bar{C}_b := W_b - C_b \neq \varnothing$ holds, for each $b \in B$, then we can define its $\mathcal{H}$-based *complement formula* $\bar{C} := \bigcup_{b \in B} \bar{C}_b = K_\mathcal{H} - C$ with fibres $\bar{C}_b$. For example, given $\mathcal{H} = (V, B)$ with $V = \{x_1, x_2, x_3\}$, and $B = \{x_1x_2, x_1x_3\}$, let $C = \{x_1\bar{x}_2, x_1x_2, x_1\bar{x}_3, \bar{x}_1\bar{x}_3\}$ then $K_\mathcal{H} = C \cup \bar{C}$ where $\bar{C} = \{\bar{x}_1x_2, \bar{x}_1\bar{x}_2, x_1x_3, \bar{x}_1x_3\}$. A *fibre-transversal (f-transversal)* of $K_\mathcal{H}$ (not to be confused with a hitting set) is a $\mathcal{H}$-based formula $F \subset K_\mathcal{H}$ such that $|F \cap W_b| = 1$, for each $b \in B$. Hence F is a formula containing exactly one clause of each fibre W_b of $K_\mathcal{H}$; let that clause be refered to as $F(b)$. For convenience let $\mathcal{F}(K_\mathcal{H})$ be the set of all f-transversals of $K_\mathcal{H}$. An important type of f-transversals F are those containing each variable of V as a pure literal, that is, occurring in F with a single polarity only. Such f-transversals are called *compatible* and have the property that $\bigcup_{b \in B} F(b) \in W_V$. Let $\mathcal{F}_{\text{comp}}(K_\mathcal{H})$ be the collection of all compatible f-transversals of $K_\mathcal{H}$. As a simple example for a compatible f-transversal, consider the base hypergraph $\mathcal{H} = (V, B)$ with variable set $V := \{x_1, x_2, x_3\}$ and $B :=$

$\{b_1 := x_1x_2,\ b_2 := x_1x_3,\ b_3 := x_2x_3\}$. Then, e.g., the clauses $c_1 := \bar{x}_1x_2 \in W_{b_1}$, $c_2 := \bar{x}_1\bar{x}_3 \in W_{b_2}$ and $c_3 := x_2\bar{x}_3 \in W_{b_3}$, denoted as literal strings, form a compatible f-transversal of the corresponding $K_{\mathcal{H}}$, because $c_1 \cup c_2 \cup c_3 = \bar{x}_1x_2\bar{x}_3 \in W_V$. In a certain sense orthogonal to compatible f-transversals are the *diagonal* f-transversals. By definition, a diagonal f-transversal F meets each compatible f-transversal F' of $K_{\mathcal{H}}$ in at least one clause; formally: for each $F' \in \mathcal{F}_{\text{comp}}(K_{\mathcal{H}})$ we have $F \cap F' \neq \varnothing$. Let $\mathcal{F}_{\text{diag}}(K_{\mathcal{H}})$ be the collection of all diagonal f-transversals of $K_{\mathcal{H}}$.

As for the total clause set $K_{\mathcal{H}}$ we can define f-transversals for a $\mathcal{H}$-based formula $C \subset K_{\mathcal{H}}$: An f-transversal F of C contains exactly one clause of each fibre C_b of C. The collection of all f-transversals of C is denoted as $\mathcal{F}(C)$. We also define compatible and diagonal f-transversals of C via $\mathcal{F}_{\text{comp}}(C) := \mathcal{F}(C) \cap \mathcal{F}_{\text{comp}}(K_{\mathcal{H}})$, and $\mathcal{F}_{\text{diag}}(C) := \mathcal{F}(C) \cap \mathcal{F}_{\text{diag}}(K_{\mathcal{H}})$.

The following result characterizes satisfiability of a formula C in terms of compatible f-transversals in its based complement formula $\bar{C}$ (cf. [12]):

Theorem 1. *For $\mathcal{H} = (V, B)$, let $C \subset K_{\mathcal{H}}$ be a $\mathcal{H}$-based formula such that $\bar{C}$ is $\mathcal{H}$-based, too. Then C is satisfiable if and only if $\bar{C}$ admits a compatible f-transversal, i.e. $\mathcal{F}_{\text{comp}}(\bar{C}) \neq \varnothing$.*

Proof. Suppose C is satisfiable and let $t \in W_V$ be one of its models. Then for each base point $b \in B = B(C) = B(\bar{C})$, the restriction $t|_b$ of t to b satisfies all clauses of the fibre W_b of $K_{\mathcal{H}}$ except for the clause $t^\gamma|_b$ obtained from $t|_b$ via complementing all literals. Hence $F_t(b) := t^\gamma|_b$ is a member of $\bar{C}$ and therefore $F_t := \{F_t(b) | b \in B\}$ is a compatible f-transversal of $\bar{C}$ because $\bar{C}$ is $\mathcal{H}$-based and $\bigcup F_t = \bigcup_{b \in B} t^\gamma|_b = t^\gamma$.

Conversely, let F be a compatible f-transversal of $\bar{C}$. Then, by definition of compatibility and because V has no isolated variables $t := \bigcup_{b \in B} F(b) \in W_V$ is a truth assignment. And we claim that via complementing all assignments we obtain a model t^γ of C. Indeed, suppose the contrary, meaning that there is a base point b and a clause c over b belonging to C that is not satisfied by t^γ. Then this clause must have the form $c = t|_b \in C$, but this means a contradiction to $F(b) = t|_b \in \bar{C}$ as F was assumed to be an f-transversal of $\bar{C}$. □

Whereas compatible f-transversals always exist, it is not clear whether diagonal transversals exist in $K_{\mathcal{H}}$. However, if there are diagonal transversals, then each fixed compatible transversal meets all diagonal transversals in $K_{\mathcal{H}}$. We have some more easy observations regarding f-transversals.

Proposition 1. *(1) $\mathcal{F}_{\text{comp}}(K_{\mathcal{H}}) \cong W_V$ (means isomorphism),*
(2) $\mathcal{F}_{\text{diag}}(K_{\mathcal{H}}) = \{F \in \mathcal{F}(K_{\mathcal{H}}) | \forall t \in W_V \exists b \in B : F(b) = t|_b\}$,
(3) $\mathcal{F}_{\text{comp}}(K_{\mathcal{H}})^\gamma = \mathcal{F}_{\text{comp}}(K_{\mathcal{H}})$,
(4) $\mathcal{F}_{\text{diag}}(K_{\mathcal{H}})^\gamma = \mathcal{F}_{\text{diag}}(K_{\mathcal{H}})$,
(5) $\mathcal{F}_{\text{diag}}(K_{\mathcal{H}}) \cap \mathcal{F}_{\text{comp}}(K_{\mathcal{H}}) = \varnothing$,
(6) $F \in \mathcal{F}_{\text{diag}}(K_{\mathcal{H}}) \Leftrightarrow F \in$ UNSAT,
(7) $F \in \mathcal{F}_{\text{comp}}(K_{\mathcal{H}}) \Rightarrow F \in$ SAT.

Proof. Assertion (1) follows from Theorem 1. Assertion (2) says that each diagonal f-transversal meets each clause in W_V and thus follows from (1) immediately.

Assertion (3) is obvious. Let $F \in \mathcal{F}_{\mathrm{diag}}(K_{\mathcal{H}})$ and assume there is $F' \in \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$ such that $F'(b) \neq F^{\gamma}(b)$, for all $b \in B$, equivalent to $F'^{\gamma}(b) \neq F(b)$, for all $b \in B$, contradicting that F is diagonal, so yielding (4). Assume there is $F \in \mathcal{F}_{\mathrm{diag}}(K_{\mathcal{H}}) \cap \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$, then with (3) also $F^{\gamma} \in \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$ holds, but we have $F^{\gamma}(b) \neq F(b)$, for each $b \in B$, therefore $F \notin \mathcal{F}_{\mathrm{diag}}(K_{\mathcal{H}})$ yielding a contradiction implying (5). According to Theorem 1, we have $F \in \mathrm{UNSAT}$ iff $\mathcal{F}_{\mathrm{comp}}(\bar{F}) = \varnothing$ iff, $\forall F' \in \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$, there is $b \in B$ such that $F'(b) = F(b) \in F$ iff $F \in \mathcal{F}_{\mathrm{diag}}(K_{\mathcal{H}})$, hence (6). (7) is implied by (6) according to (5); moreover for $F \in \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$, $\bigcup F \in W_V$ specifically satisfies F. □

Thus, we have three types of f-transversals composing $\mathcal{F}(K_{\mathcal{H}})$, namely compatible f-transversals which are always satisfiable, diagonal ones which (which do not exist in each case but) are always unsatisfiable, and finally, f-transversals that neither are compatible nor diagonal but are always satisfiable.

Definition 1. *A formula $D \subseteq K_{\mathcal{H}}$ is called a* diagonal *formula if, for each $F \in \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$, $F \cap D \neq \varnothing$ holds.*

Obviously each $F \in \mathcal{F}_{\mathrm{diag}}(K_{\mathcal{H}})$ (if existing) is a diagonal formula. Since a diagonal formula D contains a member of each compatible f-transversal the complement formula $\bar{D}$ cannot have a compatible f-transversal. Therefore $D \in \mathrm{UNSAT}$ according to Theorem 1, and we have:

Proposition 2. *A formula is unsatisfiable iff it contains a diagonal subformula.*

Consider a simple application of the concepts above: Recall that a hypergraph is called *Sperner* if no hyperedge is contained in another hyperedge [2]. Similarly, we call a formula $C \in \mathrm{CNF}$ *simple* if no clause is contained in another one. A non-simple formula C can easily be turned into a SAT-equivalent simple one by removing each clause $c' \in C$ that properly contains another clause. If C is simple its base hypergraph $\mathcal{H}(C) = (V(C), B(C))$ can either be Sperner or non-Sperner. Assuming $\mathcal{H}(C) = \mathcal{H}(\bar{C})$ we have that $\mathcal{H}(C)$ Sperner implies that $\bar{C}$ is simple. The case that $\mathcal{H}(C) = \mathcal{H}(\bar{C})$ is non-Sperner, but both C and $\bar{C}$ are simple is illustrated by the following example (where clauses and edges are represented as strings):

$$\begin{aligned} C &= \{xy,\ x\bar{y}z,\ \bar{x}yz,\ \bar{x}\bar{y}z,\ x\bar{y}\bar{z},\ \bar{x}y\bar{z},\ \bar{x}\bar{y}\bar{z}\} \\ \bar{C} &= \{x\bar{y},\ \bar{x}y,\ \bar{x}\bar{y},\ xyz,\ xy\bar{z}\} \\ B(C) &= \{xy,\ xyz\} \end{aligned}$$

Theorem 2. *Let $C \in \mathrm{CNF}$ be such that $\mathcal{H}(C) = \mathcal{H}(\bar{C})$ is non-Sperner, but both C and $\bar{C}$ are simple. Then C and $\bar{C}$ are unsatisfiable.*

Proof. For proving that C is unsatisfiable, it is sufficient to show that $\bar{C}$ cannot have a compatible f-transversal according to Theorem 1. Since $\mathcal{H}(C)$ is non-Sperner there are $b, b' \in B(C)$ with $b \subset b'$. Now, for each f-transversal $F \in \mathcal{F}(\bar{C})$, we have $F(b) \neq F(b')$, and $F(b) \not\subseteq F(b')$ because $\bar{C}$ is assumed to be simple. That means there is $x \in b$ such that $x \in F(b)$, $\bar{x} \in F(b')$ or vice versa, hence $F(b) \cup F(b') \supset \{x, \bar{x}\}$ is not compatible implying that $C \in \mathrm{UNSAT}$. By exchanging the roles of C and $\bar{C}$ we also obtain that $\bar{C}$ is unsatisfiable. □

Corollary 1. *If C is simple and satisfiable then either*
(i) $\mathcal{H}(C)$ is Sperner and $\bar{C}$ is simple or
(ii) neither $\mathcal{H}(C)$ is Sperner nor $\bar{C}$ is simple and, for each pair $b_1 \subset b_2 \in B(C)$, there are $c_1 \subset c_2 \in \bar{C}$ such that $V(c_i) = b_i$, $i = 1, 2$.

The criterion in (ii) above is not sufficient for satisfiability of C: Let $b_1 \subset b \in B(C)$ such that $c_1 \subset c \in \bar{C}$ and moreover let $b'_1 \subset b' \in B(C)$ such that $c'_1 \subset c' \in \bar{C}$, where $V(c) = b$, $V(c') = b'$, $V(c_1) = b_1$, and $V(c'_1) = b'_1$. Now assume that $b \cap b' \neq \varnothing$, and that c, c' are the only clauses over b, b' in $\bar{C}$. Clearly, if $c|_{b\cap b'} \neq c'|_{b\cap b'}$ then there is no compatible f-transversal of $\bar{C}$, so C has no model.

4 Formulas over Exact Linear Base Hypergraphs

Returning to the general discussion, let $\mathcal{H} = (V, B)$ be a non-empty base hypergraph, then clearly $\mathcal{F}_{\text{comp}}(K_{\mathcal{H}})$ is non-empty we even have $|\mathcal{F}_{\text{comp}}(K_{\mathcal{H}})| = 2^{|V|}$ according to Prop. 1 (1). However, a priori it is not clear whether in general $\mathcal{F}_{\text{diag}}(K_{\mathcal{H}}) \neq \varnothing$ holds, too. Actually, this depends on the structure of the base hypergraph $\mathcal{H}$. To that end, let us consider an interesting and guiding example regarding satisfiability of certain formulas over an *(exact) linear* base hypergraph $\mathcal{H} = (V, B)$. By definition, $\mathcal{H}$ linear has the property $|b \cap b'| \leq 1$, for all distinct $b, b' \in B$, and the exact linear case is defined replacing $\leq$ with $=$. Recall that a hypergraph $\mathcal{H} = (V, B)$ is called *loopless* iff $|b| \geq 2$, for all $b \in B$. In [14] *(exact) linear* formulas are discussed in more detail. In an (exact) linear formula the variable sets of distinct clauses have at most (resp. exactly) one member in common.

Lemma 1. *[14] Each exact linear formula without unit clauses is satisfiable.*

From the Lemma we conclude that if the base hypergraph $\mathcal{H} = (V, B)$ is exact linear and loopless then for the corresponding total clause set $\mathcal{F}_{\text{diag}}(K_{\mathcal{H}}) = \varnothing$ holds. Indeed, no unsatisfiable f-transversal can exist then, because each is exact linear and we are done by contraposing Proposition 1 (6). So indeed there are hypergraphs admitting no diagonal f-transversal. The reverse question, namely are there hypergraphs at all such that the total clause sets have diagonal f-transversals, also is answered positive: Each *unsatisfiable* linear formula obviously is an f-transversal of the total clause set over the underlying linear base hypergraph, so it is diagonal.

Fact 1. *The notion of (diagonal) f-transversals generalizes the notion of (unsatisfiable) linear formulas.*

We now address the class of all CNF formulas over exact linear base hypergraphs, called *exact linearly-based formulas*, for short. It is easy to see that this class corresponds to the class of CNF formulas such that the variable sets of *each pair* of clauses have exactly one or all members in common. Clearly, each exact linear formula also is exact linearly-based. In the following we investigate some aspects of SAT regarding the class of exact linearly-based formulas.

To a $\mathcal{H}$-based formula C, for arbitrary $\mathcal{H}$, we can assign its *fibre graph* $G(C)$ as follows: Each clause c of C corresponds to a vertex. And vertices $c, c' \in C$ form an edge iff (1) they belong to distinct fibres of C, i.e. there are $b, b' \in B$, $b \neq b'$, such that $c \in C_b$, $c' \in C_{b'}$; and (2) $c \cap c' \neq \varnothing$. In terms of the fibre graph we obtain the following characterization of satisfiability in the case of exact linear bases. Recall that a clique is a (sub)graph such that each pair of its vertices are joined by an edge.

Proposition 3. *Let C be exact linearly-based such that $\mathcal{H}(C) = \mathcal{H}(\bar{C}) =: \mathcal{H} = (V, B)$. Then C is satisfiable iff $G(\bar{C})$ admits a clique of size $|B|$.*

Proof. In view of Theorem 1 we prove that $\bar{C}$ admits a compatible f-transversal if and only if $G(\bar{C})$ has a clique of size $|B|$. Assume $G(\bar{C})$ has such a clique F, then it contains exactly one member of each of the fibres of $\bar{C}$, hence the clauses corresponding to the vertices in F form an f-transversal of $\bar{C}$, also denoted as F. Suppose that F is not a compatible f-transversal. Then the union of all clauses in F contains a variable x and its negation $\bar{x}$, i.e. a complemented pair. As by definition no clause of C has a complemented pair, there are two clauses c and c' in F such that $x \in c$ and $\bar{x} \in c'$. Since $\mathcal{H}$ is linear and F is an f-transversal $V(c)$ and $V(c')$ do not have another variable in common than x meaning $c \cap c' = \varnothing$. Because c and c' form an edge in $G(\bar{C})$ we have $c \cap c' \neq \varnothing$ yielding a contradiction, so F is a compatible f-transversal.

Conversely, assume that $\bar{C}$ has a compatible f-transversal F. Then it has size $|B|$ because $\bar{C}$ is a $\mathcal{H}$-based formula by assumption. Suppose there are two members c, c' in F whose vertices do not form an edge in $G(\bar{C})$. Then $c \cap c' = \varnothing$ implying $V(c) \cap V(c') = \varnothing$ contradicting exact linearity of $\mathcal{H}$. In conclusion, the members of F correspond to a $|B|$-clique in $G(\bar{C})$. □

Note that if $\bar{C}$ is not a $\mathcal{H}$-based formula then C is unsatisfiable trivially because it contains a complete W_b as fibre subformula. Moreover notice that the proof above is not valid for arbitrary base hypergraphs. The next observation concerns rather specific exact linearly-based formulas:

Lemma 2. *Let $\mathcal{H} = (V, B)$ be an exact linear base hypergraph such that there is a vertex $x \in V$ occuring in each $b \in B$. Let $C \subset K_{\mathcal{H}}$ be a $\mathcal{H}$-based formula such that $\bar{C}$ also is $\mathcal{H}$-based. Then we have: $C \in$ UNSAT if and only if*

$$(*) \quad |\{b \in B | \forall c \in \bar{C}_b : x \in c\}| > 0 \quad \text{and} \quad |\{b \in B | \forall c \in \bar{C}_b : \bar{x} \in c\}| > 0$$

Moreover, assuming that C is given in terms of its fibre subformulas we can check it for satisfiability in linear time $O(\|C\|)$.

Proof. In view of Prop. 3 it is sufficient to show that $G(\bar{C})$ admits no $|B|$-clique iff $(*)$ in the assertion is true. But this is obvious because $(*)$ holds for $\bar{C}$ iff there is a fibre subformula $\bar{C}_b$ containing x as pure literal, and (at least) one other $\bar{C}_{b'}$ containing $\bar{x}$ as pure literal. Hence no clause contained in $\bar{C}_b$ can be joined in $G(\bar{C})$ to a clause in $\bar{C}_{b'}$ and we proved the first assertion.

For verifying the second assertion, we decribe a simple algorithm checking $(*)$, for $\bar{C}$, via inspecting C in linear time. To that end initialize two counters

$N_x \leftarrow 0$ and $N_{\bar{x}} \leftarrow 0$. Then check each fibre subformula $C_b \subset C$, $b \in B$, whether it contains exactly $2^{|b|-1}$ clauses containing literal x, respectively literal $\bar{x}$. If the first holds true then increase counter $N_{\bar{x}}$ by one; if the second is true then increase counter N_x by one; otherwise do not modify the counters, respectively. Proceed in this way until either $N_{\bar{x}} > 0$, and $N_x > 0$ then stop, and return $C \in$ UNSAT. Or the whole formula has been inspected and $N_{\bar{x}} = 0 = N_x$ then return $C \in$ SAT. The algorithm works correctly, as it is easy to see that $(*)$ holds iff $N_{\bar{x}} > 0$ and $N_x > 0$. □

The next algorithm, instead of searching $G(\bar{C})$, works via inspecting C itself:

Theorem 3. *For C exact linearly-based and represented in terms of its fibre subformulas,* SAT *can be decided in polynomial time.*

Proof. For C exact linearly-based, by definition, there is an exact linear base hypergraph $\mathcal{H} = (V, B)$ such that $C \subseteq K_{\mathcal{H}}$. If $\mathcal{H}$ is not loopless, we are done by Lemma 2. If C is exact linear and loopless, we are done according to Lemma 1 in linear time, by inspection of the input. If $\bar{C}$ is not $\mathcal{H}$-based then C cannot be satisfiable because it contains at least one complete fibre W_b of $K_{\mathcal{H}}$. So assume that $\bar{C}$ is $\mathcal{H}$-based. The basic idea is as follows: We say that a clause c of the input formula C over n variables *meets* a truth assignment $t \in W_V$ iff $c \subseteq t$. A single clause c obviously meets exactly $2^{n-|c|}$ truth assignments. Suppose we are able to calculate fast the total number N of all distinct truth assignments met by all clauses of the input formula C. Then we have $C \in$ SAT iff $N < 2^n$. Indeed, only in that case C does not contain a diagonal formula since the $\mathcal{H}$-based complement formula $\bar{C}$ of C admits a compatible f-transversal corresponding to a truth assignment not met by any $c \in C$ according to Prop. 1. So, we are done refering to Theorem 1.

To that end, recall that $C_b \subset W_b$ denotes the fibre subformula of C over $b \in B$. Let l be an arbitrary literal occurring in the input formula C, and recall that $C_b(l) \subset C_b$ is the subformula of C_b of all clauses containing literal l, where $V(l) \in b$. The clauses in $C_b(l)$ in total meet exactly $\mu(l, b) := |C_b(l)| \cdot 2^{n-|b|}$ of all 2^{n-1} truth assignments containing l. The determination of the number of truth assignments met by the clauses in the input formula C is organized as follows: First we compute a variable x called a *maximum variable* and the corresponding edges $b_x, b_{\bar{x}} \in B$ (smallest indices if ambigous) satisfying

$$\mu(x, b_x) + \mu(\bar{x}, b_{\bar{x}}) = \max\{\mu(y, b) + \mu(\bar{y}, b') | y \in V, b, b' \in B\}$$

Note that $|C_b(l)| = 0$ if $l \notin L(C_b)$, specifically if $V(l) \notin b$ meaning $b \notin B(x)$. (Here, we have $B(x) = \{b \in B | x \in b\}$regarding B as a positive monotone clause set.) It is possible that $b_x = b_{\bar{x}}$. Next we perform at most two independent runs of a Procedure `ComputeCoverNumber`(l, p). The first one for $l = x$ and a second one for $l = \bar{x}$; returning in p the number of all l-containing truth assignments, that are met by the clauses of C containing l. Each of these executions of Procedure `ComputeCoverNumber`(l, p) is initiated only if $\mu(l, b_l) < 2^{n-1}$ meaning that the fibre subformulas corresponding to the maximum variable x do not meet all 2^{n-1} possible l-containing truth assignments. Clearly, the runs of the procedure for x and $\bar{x}$ can be processed independently. Finally, the numbers of truth assignments

met that are returned in p are added, and the algorithm returns `unsatisfiable` iff the total value equals 2^n.

Procedure `ComputeCoverNumber`(l, p) consists of two subprocedures. The first one is entered only if there is at least one fibre subformula C_b with $x \in b$ besides C_{b_l} in which literal l occurs. The subprocedure then computes all additional l-containing truth assignments met by the clauses in all these fibre subformulas. The second subprocedure is devoted to determine all further truth assignments containing l met by the remaining fibre subformulas C_b with $x = V(l) \notin b$. Similarly, it is entered only if there is at least one fibre subformula C_b with $b \in B - B(x)$.

The first subprocedure of `ComputeCoverNumber` proceeds as follows: W.l.o.g. let $\mathcal{C}_l := \{C_{b_1}, C_{b_2}, \ldots, C_{b_s}\}$, for $s \geq 1$, be the collection of all remaining fibre subformulas with $x = V(l) \in b_i$, and $|C_{b_i}(l)| > 0$, for all $1 \leq i \leq s$, where $b_1 := b_l$. Obviously, a fibre formula C_b with $b \in B(x)$ but $C_b(l) = \varnothing$ cannot contribute to the set of met l-containing truth assignments. Let $m_j := |C_{b_j}(l)|$ and $m'_j := |W_{b_j}(l) - C_{b_j}(l)| = 2^{|b_j|-1} - m_j$, for $1 \leq j \leq s$. Note that by assumption we have $m'_j > 0$, for $1 \leq j \leq s$. Now we claim that the number α_l of l-containing truth assignments met by the clauses of the subformulas in $\mathcal{C}_l$ is given by:

$$(*) \quad \alpha_l(s) := \sum_{j=1}^{s} \left[m_j \cdot 2^{n+(j-1)-\sum_{q=1}^{j} |b_q|} \cdot \prod_{k=1}^{j-1} m'_k \right]$$

where as usual $\prod_{k=1}^{j-1} m'_k := 1$ if $j = 1$. We prove the claim by induction on $s := |\mathcal{C}_l| \geq 1$. If $s = 1$ then clearly $\alpha_l(1) = \mu(l, b_l)$ and $(*)$ can easily be verified to be correct. So, let $s \geq 1$ and assume that $(*)$ is true for all values not greater than s. Let $\mathcal{C}_l$ have cardinality $s + 1$, then the first s members of $\mathcal{C}_l$ meet $\alpha_l(s)$ distinct l-containing truth assignments by the induction hypothesis.

All clauses in $C_{b_{s+1}}(l)$ contain literal l and literals over the same $|b_{s+1}| - 1$ variables that do not occur in any other b_j, $1 \leq j \leq s$, because $\mathcal{H}$ is exact linear. Let $\Delta_l(s+1)$ be the number of additional l-containing truth assignments met by the clauses in $C_{b_{s+1}}(l)$ but not by those in $\mathcal{C}_l - \{C_{b_{s+1}}(l)\}$. Further, each clause of $C_{b_{s+1}}(l)$ contributes the same number R of additional l-containing truth assignments. This is true because these clauses meet pairwise distinct parts over the range $b_{s+1} - \{x\}$ of truth assignments not already met by the clauses in $\mathcal{C}_l - \{C_{b_{s+1}}(l)\}$; we thus have $\Delta_l(s+1) = m_{s+1} \cdot R$.

To determine R we need the number of all l-containing truth assignments not already met by the clauses in $\mathcal{C}_l - \{C_{b_{s+1}}(l)\}$. Consider arbitrary clauses $c_i \in W_{b_i}(l) - C_{b_i}(l)$, $1 \leq i \leq s$, i.e. the complements of the fibre subformulas in $\mathcal{C}_l - \{C_{b_{s+1}}(l)\}$. The union $d := \bigcup_{i=1}^{s} c_i$ yields a literal set of cardinality $|d| = 1 + \sum_{i=1}^{s}(|b_i| - 1)$ which clearly cannot be contained in any of the l-containing truth assignments met by the clauses in $\mathcal{C}_l - \{C_{b_{s+1}}(l)\}$. Hence a clause $c \in C_{b_{s+1}}(l)$ meets each l-containing truth assignment enlarging a literal string d as constructed above with c. Of such a truth assignment consequently then $r := 1 + \sum_{i=1}^{s}(|b_i| - 1) + |b_{s+1}| - 1$ positions are already fixed yielding 2^{n-r} distinct such truth assignments containing that literal string d and c. Clearly, we can construct $m'_1 \cdot m'_2 \cdots m'_s$ distinct literal strings d as above. Each yielding

2^{n-r} truth assignments met by each fixed clause of $C_{b_{s+1}}(l)$. So we obtain $R = m'_1 \cdot m'_2 \cdots m'_s \cdot 2^{n-r}$ and therefore

$$\Delta_l(s+1) = m_{s+1} \cdot 2^{n+s-\sum_{q=1}^{s+1}|b_q|} \cdot \prod_{k=1}^{s} m'_k$$

additional l-containing truth assignments. Now we conclude by induction

$$\begin{aligned}\alpha_l(s+1) &= \Delta_l(s+1) + \alpha_l(s)\\ &= m_{s+1} \cdot 2^{n+s-\sum_{q=1}^{s+1}|b_q|} \cdot \prod_{k=1}^{s} m'_k + \\ &\quad + \sum_{j=1}^{s}\left[m_j \cdot 2^{n+(j-1)-\sum_{q=1}^{j}|b_q|} \cdot \prod_{k=1}^{j-1} m'_k\right]\\ &= \sum_{j=1}^{s+1}\left[m_j \cdot 2^{n+(j-1)-\sum_{q=1}^{j}|b_q|} \cdot \prod_{k=1}^{j-1} m'_k\right]\end{aligned}$$

in hamony with $(*)$, for $s+1$, finishing the proof of the claim.

Clearly, number α_l can be determined performing a simple loop recalling that by assumption $m_j > 0$, for all $1 \le j \le s$:

$z \leftarrow m_1 \cdot 2^{n-|b_1|}$
$p \leftarrow z$
for $j = 1$ **to** $s-1$ **do**
  $z \leftarrow z \cdot m'_j \cdot \frac{m_{j+1}}{m_j} \cdot 2^{1-|b_{j+1}|}$
  $p \leftarrow p + z$
do

So finally, we have to check whether $p = 2^{n-1}$. In order to avoid calculations with possibly large number 2^n it is sufficient instead to compute $p' := p/2^n$. That means to start with $z \leftarrow m_1 \cdot 2^{-|b_1|}$ and finally to check whether $p' = 1/2$.

Recall that the second subprocedure of `ComputeCoverNumber`(p, l) is devoted to determine all additional l-containing truth assignments met by all clauses in fibre subformulas C_b with $b \in B - B(x)$ meaning $x = V(l) \notin b$. Clearly this subprocedure needs to be started only if $p' < 1/2$, because otherwise all l-containing truth assignments are met already. For explaining the second subprocedure, let c' be a clause of a fibre subformula $C_{b'}$ with $b' \in B - B(x)$. We claim that c' meets a *not yet encountered* l-containing truth assignment if and only if for each $b \in B(x)$ there is $c \in W_b(l) - C_b(l)$ with $c' \cap c \neq \emptyset$. To prove the claim, recall that by assumption we have $m'_b = |W_b(l) - C_b(l)| > 0$ for all $b \in B(x)$ since otherwise $\mu(l, b_l) = 2^{n-1}$ and Procedure `ComputeCoverNumber`(p, l) would not have been entered at all. So, there always is at least one selection $S := \{c \in W_b(l) - C_b(l) | b \in B(x)\}$. As above we build the literal string $d = \bigcup S$ satisfying $|d| = 1 + \sum_{b \in B(x)}(|b| - 1)$ and $|V(c') \cap V(d)| = |S|$, for the chosen clause c', because of exact linearity. None of the clauses in $\mathcal{C}_l$ can meet any l-containing truth

assignment enlarging d. Clearly c' meets such a truth assignment iff $|c' \cap d| = |S|$ which is equivalent to $c' \cap c \neq \emptyset$ for all $c \in S$ proving the claim.

W.l.o.g. let $\mathcal{C} := \{C_{b_{s+1}}, \ldots, C_{b_{s+r}}\}$ with $r := |\mathcal{C}| \geq 1$, be the collection of all fibre subformulas of C neither containing x nor $\bar{x}$. For $c' \in C_{b_{s+1}}$, let $\{y_i\} = V(c') \cap b_i$ which, for all $1 \leq i \leq s$, are uniquely determined and pairwise distinct because of exact linearity, hence $|c'| \geq s$. Assume that $l_i \in c'$ is the corresponding literal with $V(l_i) = y_i \neq x$. Let $n_l := \sum_{q=1}^{s} |b_q| - (s-1)$ be the number of distinct variables in $V(\mathcal{C}_l)$. Let $\lambda_i(c')$ be the number of clauses in $W_{b_i}(l) - C_{b_i}(l)$ containing literal l_i, $1 \leq i \leq s$. Clearly, l_i is contained in exactly $2^{|b_i|-2}$ clauses of $W_{b_i}(l)$. So, if l_i occurs t_i times in $C_{b_i}(l)$ we have $\lambda_i(c') = 2^{|b_i|-2} - t_i$, for $1 \leq i \leq s$. Since each clause in $C_{b_{s+1}}$ already fixes $n_l + (|b_{s+1}| - s)$ of n positions, we conclude with the claim above that all clauses in $C_{b_{s+1}}$ together meet exactly

$$2^{n-n_l-(|b_{s+1}|-s)} \sum_{c \in C_{b_{s+1}}} \prod_{j=1}^{s} \lambda_j(c)$$

additional l-containing truth assignments.

On that basis we obtain via induction on $r = |\mathcal{C}| \geq 1$ analogous to the argumentation for the first subprocedure, that all members of $\mathcal{C}$ together meet

$$\sum_{k=1}^{r} \left[2^{n-n_l-\sum_{j=1}^{k} f(j)} \sum_{c \in C_{b_{s+k}}} \left(\prod_{j=1}^{s+k-1} \lambda_j(c) \right) \right]$$

many additional distinct l-containing truth assignments. Here, for $1 \leq j \leq r$,

$$f(j) := |b_{s+j}| - \left| \bigcup_{i=1}^{s+j-1} (b_{s+j} \cap b_i) \right| \in \{0, \ldots, |b_{s+j}| - s\}$$

is the number of variables remaining from b_{s+j} if each variable contained in one of $\{b_1, \ldots, b_{s+j-1}\}$ is removed. Hence, $n_l + \sum_{j=1}^{k} f(j) = |V(\{b_1, \ldots, b_{s+k}\})|$.

Since the formula is represented through its fibres $C_b, b \in B$, and $|B| \leq V(C)$ because of exact linearity [14], all needed values can be collected in polynomial time. □

The method above can easily be adapted to solve the counting problem #-SAT for exact linearly-based formulas in polynomial time: execute both subprocedures yielding the counts N_1, N_2 respectively, then return $2^n - (N_1 + N_2)$. Further, a poly-time algorithm for the search problem can be provided by self-reduction: Iteratively set a variable and check by the algorithm above, whether the resulting formula remains satisfiable. In the negative case, fix the selected variable complementary, etc. Though our class is not stable under partial assignments, the algorithm above still works since the underlying hypergraph shrinks.

Moreover notice that checking whether the subformula $\{C_b | b \in B(x)\}$ of C already is unsatisfiable could be done fast according to Lemma 2, instead of running the first subprocedure twice. However if that does not yield unsatisfiability,

in general we cannot simply proceed with the second subprocedure because we do not know how many compatible f-transversals have been met by the clauses in the fibre subformulas over $B(x)$.

5 The Fibre View Further Exploited

Next we investigate some other formula classes using the fibre view concept, besides exact linearly-based formulas. For $\mathcal{H} = (V, B)$, let $C \subset K_{\mathcal{H}}$ such that $B(C) = B = B(\bar{C})$. If $C \in \mathrm{SAT}$ then according to Prop. 1 (1) for each $t \in M(C)$ there is a unique $F \in \mathcal{F}_{\mathrm{comp}}(K_{\mathcal{H}})$ with $\bigcup F = t$. We now address the question in which case each model t of C corresponds to an $F \in \mathcal{F}_{\mathrm{comp}}(C)$, i.e. corresponds to a compatible f-transversal of the *formula itself.*

Lemma 3. *If $C \in \mathrm{CNF} \cap \mathrm{SAT}$, $B(C) = B(\bar{C})$, such that for each $t \in M(C)$ there is $F \in \mathcal{F}_{\mathrm{comp}}(C)$ with $\bigcup F = t$. Then $\bar{C} \in \mathrm{SAT}$ and $\bigcup F' \in M(\bar{C})$ for each $F' \in \mathcal{F}_{\mathrm{comp}}(\bar{C})$; and vice versa.*

Proof. Let $F \in \mathcal{F}_{\mathrm{comp}}(\bar{C})$ and $t := \bigcup F \in W_V$, then according to the proof of Theorem 1, t^γ is a model of C. By assumption there is $F' \in \mathcal{F}_{\mathrm{comp}}(C)$ such that $t^\gamma = \bigcup F'$. Hence, again by Theorem 1, t is a model of $\bar{C}$ as claimed, specifically $\bar{C} \in \mathrm{SAT}$. For the vice versa assertion exchange the roles of C and $\bar{C}$. □

Next we provide a formula class satisfying the assumption of the last lemma. Recall that an asymmetric formula C has the property that $c \in C$ implies $c^\gamma \notin C$ and that $\mathrm{Asym} \subseteq \mathrm{CNF}$ denotes the class of all asymmetric formulas.

Lemma 4. *Let $C \in \mathrm{CNF}$ be such that $B(C) = B(\bar{C})$ and $\bar{C} \in \mathrm{Asym}$. Then $C \in \mathrm{SAT}$ implies $\bar{C} \in \mathrm{SAT}$ and each $t \in M(C)$ corresponds to a compatible f-transversal in $\mathcal{F}_{\mathrm{comp}}(C)$; and vice versa.*

Proof. Let $t \in M(C) \neq \varnothing$ then for each $b \in B(C)$ $t|_b$ satisfies all of W_b except for $(t|_b)^\gamma$ which thus must be a clause of $\bar{C}$. And $\bar{C} \in \mathrm{Asym}$ implies that $t|_b \in C$ for each $b \in B(C)$. Hence $\{t|_b | b \in B(C)\}$ is a compatible f-transversal of C. It follows that $\bar{C} \in \mathrm{SAT}$ and that for each $t \in M(C)$ there is $F \in \mathcal{F}_{\mathrm{comp}}(C)$ with $\bigcup F = t$. For the vice versa assertion exchange the roles of C and $\bar{C}$. □

Corollary 2. *Let $C \in \mathrm{Asym}$ such that also $\bar{C} \in \mathrm{Asym}$ and $B(C) = B(\bar{C})$. Then $C \in \mathrm{SAT}$ if and only if $\bar{C} \in \mathrm{SAT}$.* □

In view of Theorem 1 we know that a formula is satisfiable if and only if the complement formula admits a compatible f-transversal. Therefore the specific class of formulas C such that every f-transversal of C is compatible is of interest, because any f-transversal gives rise to a model of $\bar{C}$ then, and vice versa. A characterization of that very specific class can be provided as follows: Let $\mathcal{H} = (V, B)$ be a base hypergraph. Then in general there is a 2-partition of V given by sets V_I and V_U. Here V_U contains all vertices occurring in only one edge $b \in B$, and V_I contains all remaining vertices occurring in at least one edge intersection.

Lemma 5. *Let* $C \in$ CNF *such that* $\mathcal{H}(C) = \mathcal{H}(\bar{C}) =: \mathcal{H} := (V, B)$. *Then* $\mathcal{F}_{\rm comp}(C) = \mathcal{F}(C)$ *iff* $(*)$*: each variable in* $V_I \subseteq V$ *is a pure literal in* C.

Proof. Assume that $(*)$ is true. Then each $y \in b \cap b'$ for all $b, b' \in B, b \neq b'$ is a pure literal in C and therefore each $F \in \mathcal{F}(C)$ is compatible. Conversely, $F \in \mathcal{F}(C)$ is compatible iff all variables are pure literals in F. Hence if each $F \in \mathcal{F}(C)$ is compatible it is easy to see that all variables occurring in edge intersections must be pure literals in C. □

Let $\mathrm{CNF}_{\rm comp}$ denote the class of all formulas $C \in$ CNF with $B(C) = B(\bar{C})$, and such that $\mathcal{F}(\bar{C}) = \mathcal{F}_{\rm comp}(\bar{C})$. As an example for $C \in \mathrm{CNF}_{\rm comp}$, let $\mathcal{H} = (V, B)$ with $V = \{q, r, s, t, u, v, x, y\}$, $B = \{b_1 = xy, b_2 = yuv, b_3 = vxr, b_4 = rst, b_5 = txq\}$, then the following formula $\bar{C}$

$$\begin{array}{llllll} \bar{C} = x\bar{y} & \bar{y}uv & vxr & rs\bar{t} & \bar{t}xq \\ & \bar{y}\bar{u}v & & r\bar{s}\bar{t} & \bar{t}x\bar{q} \end{array}$$

where clauses are arranged fibrewise obviously has the property that $\mathcal{F}(\bar{C}) = \mathcal{F}_{\rm comp}(\bar{C})$, hence $C = K_{\mathcal{H}} - \bar{C} \in \mathrm{CNF}_{\rm comp}$ implying $C \in$ SAT.

Theorem 4. *We can check in polynomial time whether an input formula* $C \in$ CNF *belongs to* $\mathrm{CNF}_{\rm comp}$. *In the positive case a model can be provided in polynomial time assuming for both that* C *is represented through its fibre subformulas.*

Proof. Let $\mathcal{H} = (V, B) = \mathcal{H}(C) = \mathcal{H}(\bar{C})$. In linear time we check whether C contains a whole W_b as fibre subformula, in which case $C \notin \mathrm{CNF}_{\rm comp}$. Otherwise, in view of Lemma 5 we have to check whether each variable in V_I is a pure literal in $\bar{C}$. First, we check in B which variables occur uniquely yielding V_U and set $V_I = V - V_U$. For each $y \in V_I$ we check whether it occurs in different polarities in $\bar{C}[B(y)] := \{\bar{C}_b | b \in B(y)\}$ simultaneously. Recall that $B(y) = \{b \in B | y \in b\}$. This test can be performed in linear time $O(\|C[B(y)]\|)$ similar to the procedure presented in the second part of the proof of Lemma 2: Initialize two counters $N_y \leftarrow 0$ and $N_{\bar{y}} \leftarrow 0$. Then check each fibre subformula C_b, $b \in B(y)$, whether it contains exactly $2^{|b|-1}$ clauses containing literal y, respectively literal $\bar{y}$. If the first holds true then increase counter $N_{\bar{y}}$ by one; if the second is true then increase counter N_y by one; otherwise do not modify the counters, respectively. Proceed in this way until either $N_{\bar{x}} > 0$ and $N_x > 0$ then stop, and return $C \notin \mathrm{CNF}_{\rm comp}$, because y occurs in both polarities (in distinct fibre subformulas). Or the whole formula has been inspected. Then we set a flag to $C \in \mathrm{CNF}_{\rm comp}$ if and only if $(*)$: $N_{\bar{y}} = 0$ and $N_y = |B(y)|$, or vice versa. The test works correctly, as it is easy to see that $(*)$ holds iff the clauses in the fibre subformulas in $\bar{C}[B(y)]$ either all contain y or $\bar{y}$. If in that way the algorithm does not return $C \notin \mathrm{CNF}_{\rm comp}$ we have checked all $y \in V_I$, and we know that $C \in \mathrm{CNF}_{\rm comp}$.

Finally, to provide a model of $C \in \mathrm{CNF}_{\rm comp}$ we simply need to select one clause $c_b \in \bar{C}_b = W_b - C_b$ for each $b \in B$ ensuring that $\bigcup_{b \in B} c_b^{\gamma} \in M(C)$ according to Theorem 1. For fixed $b = \{b_{i_1}, \ldots, b_{i_{|b|}}\}$, the selection can be performed, e.g., by ordering the members $c = \{b_{i_1}^{\varepsilon_{i_1}(c)}, \ldots, b_{i_{|b|}}^{\varepsilon_{i_{|b|}}(c)}\}$ in C_b according

to the lexicographic order of the vectors $(\varepsilon_{i_1}(c), \ldots, \varepsilon_{i_{|b|}}(c)) \in \{0,1\}^{|b|}$, and taking clause in the first gap w.r.t. all of W_b, for each $b \in B$. □

6 Concluding Remarks and Open Problems

We provided several CNF subclasses that are polynomial time decidable resp. behave trivially regarding SAT using the fibre view on clause sets. The most interesting are exact linearly-based formulas. We showed that the decision, search and counting variants of SAT restricted to this class all are polynomial time solvable. We leave it as an open problem to design a more direct algorithm for the decision, resp. search variants of SAT for exact linearly-based formulas. A future work perspective is to elaborate more deeply the relationships between the polynomial time classes studied here to other such classes. Finally, it might be of interest to relate Theorem 1 to the characterization of satisfiability of CNF formulas in the framework of binary decision diagrams.

Acknowledgement. We would like to thank the anonymous reviewers for their valuable comments.

References

1. Aspvall, B., Plass, M.R., Tarjan, R.E.: A linear-time algorithm for testing the truth of certain quantified Boolean formulas. Inform. Process. Lett. 8, 121–123 (1979)
2. Berge, C.: Hypergraphs. North-Holland, Amsterdam (1989)
3. Boros, E., Crama, Y., Hammer, P.L.: Polynomial time inference of all valid implications for Horn and related formulae. Annals Math. Artif. Int. 1, 21–32 (1990)
4. Boros, E., Hammer, P.L., Sun, X.: Recognition of q-Horn formulae in linear time. Discrete Appl. Math. 55, 1–13 (1994)
5. Franco, J., Gelder, A.v.: A perspective on certain polynomial-time solvable classes of satisfiability. Discrete Appl. Math. 125, 177–214 (2003)
6. Kleine Büning, H., Lettman, T.: Propositional logic, deduction and algorithms. Cambridge University Press, Cambridge (1999)
7. Kratochvil, J., Savicky, P., Tusa, Z.: One more occurrence of variables makes satisfiability jump from trivial to NP-complete. SIAM J. Comput. 22, 203–210 (1993)
8. Knuth, D.E.: Nested satisfiability. Acta Informatica 28, 1–6 (1990)
9. Lewis, H.R.: Renaming a Set of Clauses as a Horn Set. J. ACM 25, 134–135 (1978)
10. Minoux, M.: LTUR: A Simplified Linear-Time Unit Resolution Algorithm for Horn Formulae and Computer Implementation. Inform. Process. Lett. 29, 1–12 (1988)
11. Monien, B., Speckenmeyer, E.: Solving satisfiability in less than 2^n steps. Discrete Appl. Math. 10, 287–295 (1985)
12. Porschen, S.: A CNF Formula Hierarchy over the Hypercube. In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS (LNAI), vol. 4830, pp. 234–243. Springer, Heidelberg (2007)
13. Porschen, S., Speckenmeyer, E.: Satisfiability of Mixed Horn Formulas. Discrete Appl. Math. 155, 1408–1419 (2007)
14. Porschen, S., Speckenmeyer, E., Randerath, B.: On linear CNF formulas. In: Biere, A., Gomes, C.P. (eds.) SAT 2006. LNCS, vol. 4121, pp. 221–225. Springer, Heidelberg (2006)

15. Schaefer, T.J.: The complexity of satisfiability problems. In: Proc. STOC 1978. ACM, pp. 216–226 (1978)
16. Schlipf, J., Annexstein, F.S., Franco, J., Swaminathan, R.P.: On finding solutions for extended Horn formulas. Inform. Process. Lett. 54, 133–137 (1995)
17. Tovey, C.A.: A Simplified NP-Complete Satisfiability Problem. Discrete Appl. Math. 8, 85–89 (1984)