Unsatisfiable Linear CNF Formulas Are Large, and Difficult to Construct Explicitely

Dominik Scheder *

Theoretical Computer Science, ETH Zürich
CH-8092 Zürich, Switzerland
dscheder@inf.ethz.ch

May 11, 2009

Abstract. We call a CNF formula *linear* if any two clauses have at most one variable in common. We show that there exist unsatisfiable linear k-CNF formulas with at most $O(k^3 4^k)$ clauses, and on the other hand, any linear k-CNF formula with at most $\frac{4^k}{4e^2k^3}$ clauses is satisfiable. The upper bound uses a probabilistic construction, and we have no explicit construction coming even close to it. We give some arguments why it is difficult to find explicit constructions: First, any treelike resolution refutation of any unsatisfiable linear k-CNF formula has size at least $2^{2^{\frac{k-1}{2}-1}}$. Second, if we require the unsatisfiable linear k-CNF formula to exhibit a certain recursive structure, then we need at least $\alpha^{\alpha^{\cdots^{\alpha}}}$ clauses, where α is roughly 2 and the size of this tower is roughly k.

1 Introduction

How difficult is it to come up with an unsatisfiable CNF formula? Stupid question, of course: $\{\{x\}, \{\bar{x}\}\}$, here is one. Two clauses, each containing one literal, and unsatisfiable. Well, yes, but what if we want a *k-CNF formula*, i.e., we require that every clause contains exactly k literals? Now it is a little bit less trivial, but still easy: Take a clause $\{x_1, x_2, \ldots, x_k\}$, then $\{\bar{x}_1, x_2, \ldots, x_k\}$, $\{x_1, \bar{x}_2, \ldots, x_k\}$, until you have exhausted all 2^k combinations of negative and positive literals. Each assignment to the k variables dissatisfies exactly one clause: This formula has 2^k clauses, and it is unsatisfiable. It is extremal in the sense that all smaller k-CNF formulas are satisfiable. What if we impose further restrictions? For example, what if any two clauses must be disjoint, i.e., have no variable in common? This is too much to ask for: Such formulas are always satisfiable. Let us weaken the disjointness restriction a little bit. What if we require that any two clauses have *at most one* variable in common? This is what we call a *linear* formula. Are there unsatisfiable k-CNF formulas of that type? For which k, for all k? If yes, how do they look, how large are they? The goal of this paper is to give some answers to these questions. The class of linear formulas has already been investigated by Porschen, Speckenmeyer and Zhao [1], but that work focuses more on existence and hardness of linear formulas, and not on extremal parameters, as this paper does.

A related question, in the context of hypergraphs, has drawn some attention for quite some time already. A hypergraph is called *linear* if any two hyperedges share at most one vertex. The term *linear* probably comes from the fact that lines in a vector space intersect at most once. The hypergraph analog to our question reads as follows: Do there exist r-uniform linear hypergraphs that are not k-colorable, for any r and k? If yes,

* Research is supported by the SNF Grant 200021-118001/1

how large (in terms of the number of hyperedges) are they? This first question has been answered positively. The proof of the case $k = 2$ is due to Abbott [2]. For general k, existence follows for example from the Hales-Jewett theorem [3]. Rather tight bounds on the size of linear r-uniform non-k-colorable hypergraphs have later been given by Kostochka, Mubayi, Rödl and Tetali [4], using probabilistic techniques.

In the context of CNF formulas, a different extremal parameter has been examined. For a CNF formula F and a variable x, let $d(x, F)$ denote the number of clauses containing x or $\bar{x}$, and let $d(F) = \max_x d(x, F)$. A (k, d)-CNF formula is a k-CNF formula F such that $d(F) \leq d$. We define the extremal function $f(k)$ by

$$f(k) := \max\{d \mid \text{ every } (k,d)\text{-CNF formula is satisfiable}\} . \tag{1}$$

The function $f(k)$ has first been investigated by Tovey [5], who showed, using Hall's Theorem, that every (k, k)-CNF formula is satisfiable, thus establishing $f(k) \geq k$. Kratochvíl, Savický and Tuza [6] later proved that $f(k) \geq \frac{2^k}{ek}$, and that while $(k, f(k))$-CNF formulas are all trivially satisfiable, satisfiability of $(k, f(k)+1)$-CNF formulas is already NP-complete. For an upper bound, it is clear that $f(k) \leq 2^k - 1$, see for example the k-CNF formula we gave at the very beginning. However, better bounds are known. Hoory and Szeider [7] showed that $f(k) \in \mathcal{O}\left(\frac{\ln(k)2^k}{k}\right)$, and recently Gebauer [8] proved that $f(k) \in \mathcal{O}\left(\frac{2^{k+2}}{k}\right)$, thus $f(k)$ is now known up to a constant factor.

Returning to our question, we define:

$$m_{\text{LIN}}(k) := \max\{m \mid \text{ every linear } k\text{-CNF formula with } \leq m \text{ clauses is satisfiable}\} \tag{2}$$

For proving lower bounds on $m_{\text{LIN}}(k)$, we will actually use the lower bound on $f(k)$ given by Kratochvíl, Savický and Tuza [6]. Upper bounds on $m_{\text{LIN}}(k)$ are obtained by probabilistic constructions, not by explicit ones, as for $f(k)$. In fact, we will prove that no "straightforward" (in a sense to be made precise later) construction can yield an upper bound on $m_{\text{LIN}}(k)$ that comes even close to its true value. This, we think, highlights in some strong way the power of the probabilistic method.

The paper is organized as follows: In Section 3, we will give a randomized construction of an unsatisfiable linear k-CNF formula with roughly $O(k^3 4^k)$ clauses. We will show that this is optimal up to a factor polynomial in k. In Section 4, we will give arguments why it is probably difficult to explicitely construct "reasonably-sized" unsatisfiable linear formulas. The first argument is that when one constructs CNF formulas whose unsatisfiability is "obvious" from the way one constructs them, then most of these formulas have a small treelike resolution refutation. In contrast, we will show that any tree-like resolution refutation of any unsatisfiable linear k-CNF must have size at least $2^{2^{\frac{k}{2}}}$. Our second argument is that if we want our formula F to have a certain recursive structure (to be defined later), then F will have a "tower-like" size, i.e. at least $\alpha^{\alpha^{\cdots^{\alpha}}}$, where $\alpha > 1$ and the height of the tower is roughly k.

2 Notation

We think of a formula as a set of distinct clauses, and of a clause as a set of literals. A literal is either a variable x or its negation $\bar{x}$. We assume no clause contains both x and $\bar{x}$, for any x. In some abuse of terminology, we say a clause C contains a variable x if $x \in C$ or $\bar{x} \in C$. By $\text{vbl}(C)$, we denote the set of variables contained in C. Hence,

a formula is linear if $|\mathrm{vbl}(C) \cap \mathrm{vbl}(D)| \leq 1$ for any distinct $C, D \in F$. For a formula F and a partial assignment α to its variables, we write $F^{[\alpha]}$ to denote the formula obtained from F by removing all clauses satisfied by α, and removing all dissatisfied literals from the remaining clauses. We define the function $\mathrm{tower}_\alpha(k)$ as recursively by

$$\begin{aligned} \mathrm{tower}_\alpha(0) &= 1 \\ \mathrm{tower}_\alpha(k+1) &= \alpha^{\mathrm{tower}_\alpha(k)} \ . \end{aligned}$$

3 Existence and Upper and Lower Bounds

3.1 Existence

Theorem 3.1 ([1], [9]). *For every k, there exists an unsatisfiable linear k-CNF formula F_k containing m_k clauses, where $m_0 = 1$ and $m_{k+1} = m_k 2^{m_k}$.*

Proof. We set $F_0 = \{\square\}$. Suppose we have constructed F_k, and want to construct F_{k+1}. We create m_k new variables $x_1, \ldots, x_{m_k}$, and let $D_1, D_2, \ldots, D_{2^{m_k}}$ be all possible clauses over these variables. The formula $G := \{D_1, D_2, \ldots, D_{2^{m_k}}\}$ is unsatisfiable, but not linear. We take 2^{m_k} variable disjoint copies of F_k, denoted by $F_k^{(1)}, F_k^{(2)}, \ldots, F_k^{(2^{m_k})}$. For each $1 \leq i \leq 2^{m_k}$, we build a linear $(k+1)$-CNF formula $\tilde{F}_k^{(i)}$ from $F_k^{(i)}$ by adding, for each $1 \leq j \leq m_k$, the j^{th} literal of D_i to the j^{th} clause of $F_k^{(i)}$. Note that $\tilde{F}_k^{(i)} \equiv D_i$, i.e. they really describe the same boolean function. Finally, we set $F_{k+1} = \bigcup_{i=1}^{2^{m_k}} \tilde{F}_k^{(i)}$. This is an unsatisfiable linear $(k+1)$-CNF formula with $m_{k+1} = m_k 2^{m_k}$ clauses. □

This construction is simple, but it produces formulas of gigantic size, larger than $\mathrm{tower}_2(\mathrm{k})$. For example, printing F_4 would exceed the amount of paper available in the universe. However, unsatisfiable linear k-CNF formulas of manageable size do exist, as we shall see.

3.2 Upper Bounds

Theorem 3.2. *There exists an unsatisfiable linear k-CNF formula with at most $8k^3 4^k$ clauses.*

Proof. Take a linear k-uniform hypergraph $H = (V, E)$ with n vertices and m edges, to be determined later. By viewing the vertices as variables and hyperedges as clauses, this is a (satisfiable) linear k-CNF formula. We now replace each literal in each clause by its complement with probability $\frac{1}{2}$, independently in each clause. Let F denote the resulting (random) formula. Any fixed assignment α has a $1 - 2^{-k}$ chance of satisfying a given clause of F, and thus

$$\Pr[\alpha \text{ satisfies } F] = (1 - 2^{-k})^m < e^{-m2^{-k}} \ .$$

There are 2^n distinct assignments, hence by the union bound

$$\Pr[\text{some } \alpha \text{ satisfies } F] < 2^n e^{-m2^{-k}} = e^{\ln(2)n - m2^{-k}} \ .$$

If $m/n \geq \ln(2)2^k$, the above expression is at most 1, and hence with positive probability, no assignment satisfies F, in other words, some F is unsatisfiable.

We construct a linear k-uniform hypergraph with few hyperedges, but with a large hyperedge-vertex ratio. Let $q \in \{k, \ldots, 2k\}$ be a prime power. Choose $d \in \mathbb{N}$ such that $q^2 \ln(2) 2^k \leq q^d < q^3 \ln(2) 2^k$ and set $n := q^d$. Consider the d-dimensional vector space $\mathbb{F}_q^d$

over the field $\mathbb{F}_q$. It has n elements, called *points*. In a vector space, there is a line through any pair of points, and a line has q elements. Hence there are exactly $\binom{n}{2}/\binom{q}{2} \geq \frac{n^2}{q^2}$ lines in $\mathbb{F}_q^d$. By choice of d, we show that this lower bound is tight, up to a polynomial factor in k. The upper bound is similar to the bound for non-2-colorable linear k-uniform hypergraphs, but the proof is somewhat simpler.have $n\ln(2)2^k \leq \frac{n^2}{q^2}$, hence we can choose $m := n\ln(2)2^k$ distinct lines in $\mathbb{F}_q^d$. From each such line arbitrarily select k points and form a hyperedge. Let E be the set of all m hyperedges formed this way. The reader may check that two distinct lines cannot yield the same hyperedge. Let E be the set of these m lines obtained this way. Thus, $H = (\mathbb{F}_q^d, E)$ is a k-uniform hypergraph. It is linear, since any pair of distinct lines intersect in at most one point. By construction, $\frac{m}{n} = \ln(2)2^k$, and $m = n\ln(2)2^k \leq q^3 n\ln(2)^2 4^k \leq ln(2)^2 8k^3 4^k$, which proves the upper bound. □

3.3 Lower Bounds

One of the miracles of the probabilistic method is that, though seemingly coarse, it often leads us pretty close to the truth.

Theorem 3.3. *Any linear k-CNF formula on at most $\frac{4^k}{4e^2k^3}$ clauses is satisfiable.*

The proof of the Theorem is quite similar to the proof by Erdős and Lovász [10] of a lower bound on the size of linear hypergraphs that are not 2-colorable. We say a variable x is *frequent in F* if $d(x, F) \geq \frac{2^k}{2ek}$. Theorem 3.3 will follow easily from the following lemma.

Lemma 3.4. *Let F be a linear k-CNF formula. If there are less than $\frac{2^k}{2ek}$ variables that are frequent in F, then F is satisfiable.*

Proof. Our proof will use a result by Kratochvíl, Savicky and Tuza [6], itself a consequence of the Lovász Local Lemma. Here, a ($\geq k$)-CNF formula is a CNF formula where every clause has at least k literals.

Theorem 3.5 ([6]). *If F is a ($\geq k$)-CNF formula, and $d(x, F) \leq \frac{2^k}{ek}$ for every variable, then F is satisfiable.*

We cannot apply Theorem 3.5 directly to our linear formula F. First, we will obtain a new, "stricter" formula F' from F, by deleting certain literals from certain clauses. We can then apply Theorem 3.5 to F' to show that F' is satisfiable. Let F' be defined as follows. For each $C \in F$, we distinguish two cases. If C contains exactly one variable that is frequent in F, obtain C' from C by deleting the literal of that frequent variable. Otherwise, let C' just be C. We define $F' := \{C' \mid C \in F\}$. Observe that F' contains k-clauses as well as $(k-1)$-clauses, and if some $C \in F'$ contains a variable that is frequent in F, it contains at least two of those.

Claim: $d(x, F') \leq \frac{2^k}{ek}$, for any variable x. If variable x is not frequent in F, then the claim is trivial, since $d(x, F') \leq d(x, F)$. So suppose x is frequent in F, write $t := d(x, F')$, and let $C_1, C_2, \ldots, C_t$ be the clauses of F' containing x. By construction of F', each C_i contains, besides x, a variable y_i that is frequent in F. The y_i are all distinct, since if y_i and y_j were the same variable, then the clauses C_i and C_j would share more than one variable, namely x and y_i, contradicting the fact that F' is linear. This is the only point in the proof where we use linearity. Hence, there are at least t variables that are frequent in F, namely $y_1, y_2, \ldots, y_t$ (of course, there are even more, for example x, but let's not be picky). By assumption, there are at most $\frac{2^k}{ek}$ frequent variables in F, hence

$d(x, F') = t \leq \frac{2^k}{ek}$, which proves the claim.

Theorem 3.5, applied to F', with $k-1$ instead of k, shows that F' is satisfiable. Any assignment satisfying F' satisfies F, as well. This proves the lemma. □

Proof (of Theorem 3.3). We prove the contrapositive. Let F be an unsatisfiable linear k-CNF formula. Let U be the set of frequent variables in F. By Lemma 3.4, $|U| \geq \frac{2^k}{2ek}$. We obtain

$$k|F| = \sum_x d(x, F) \geq \sum_{x \in U} d(x, F) \geq |U| \frac{2^k}{2ek} > \left(\frac{2^k}{2ek}\right)^2 .$$

This proves the theorem. □

The trick in this proof is that by deleting certain literals in F, we seem to make our life harder, by making F "less satisfiable". However, though the new formula F' has less satisfying assignments than F, its structure makes it easier for the Lovász Local Lemma (which here is hidden in Theorem 3.5 to detect them. This might sound surprising. However, one should keep in mind that for any satisfiable formula F, there is a way to delete literals such that satisfiability becomes obvious to everybody: Just fix a satisfying assignment for F, then remove all literals not satisfied. This new formula F' only has pure literals, and its satisfiability it obvious. The point is that in the proof of Lemma 3.4, we have clear criteria which literals to delete.

4 Why are Explicit Constructions So Bad?

In the last section, we have seen simple proofs of almost matching upper and lower bounds on the size of unsatisfiable linear k-CNF formulas. This is in stark contrast to the gigantic size of the formula constructed in Theorem 3.1. In fact, we have not found any explicit construction that yields formulas that are not tower-like. In extremal combinatorics, one is used to probabilistic constructions being better than explicit ones, but this case seems extreme. In this section we will shed some light on why this is so.

To explain the difficulty of explicitely constructing unsatisfiable linear k-CNF formulas, let us make a – rather philosophical – distinction between *direct* and *indirect* constructions of CNF formulas. Later we will give a precise, albeit quite restrictive definition of direct constructions. Suppose we construct an unsatisfiable formula F from smaller ones, say $F_1, \ldots, F_m$, which we have already constructed, by modifying the F_i in some way, and then combining them to form a new formula F, in a way that makes sure that F is unsatisfiable. This we call a *direct construction*. For example, the construction in Theorem 3.1 is a direct one. Other authors, like Kratochvíl et al. [6], Hoory and Szeider [7], and, recently, Gebauer [8], who have studied the function $f(k)$ defined in (1) obtained good upper bounds using direct constructions.

By contrast, what we consider *indirect* constructions are typically used for proving lower bounds on resolution complexity. One takes a simple combinatorial theorem and translates the negation of it into a CNF formula. The theorem in question could be the pigeon hole principle (Buss and Pitassi [11]), or the fact that in a graph, the number of odd-degree vertices is even (these are called Tseitin formulas, see [12]). For a fixed number of pigeons and holes, or for a graph on a fixed number of vertices, the statement can be formulated in propositional logic, and thus resolution can be used to prove it. See for example Ben-Sasson and Wigderson [13] for lower bounds on the resolution complexity

of these formulas. In indirect constructions, unsatisfiability is clear from global "combinatorical" or "algebraic" considerations, whereas for direct constructions, it is clear from local considerations during the construction process. Consequently, directly constructed formulas typically admit short resolution proofs, or even short treelike resolution proofs. A good argument for why there are no good (direct) constructions for unsatisfiable linear k-CNF formulas would thus be to give a lower bound on their resolution complexity. This is will be our first result in this section.

Definition 4.1. *A clause C is obtained from clauses D_1, D_2 by a* resolution step *if there is a variable x such that $x \in D_1, \bar{x} \in D_2$, and $C = (D_1 \setminus \{x\}) \cup (D_2 \setminus \{x\})$. We call x the* resolved *variable and C the* resolvent *of D_1, D_2. A* resolution tree *of a formula F is a binary tree T whose vertices are labeled with clauses, such that*

- *every leaf of T is labeled with a clause in F,*
- *if a node labeled C has children labeled D_1 and D_2, then C is the resolvent of D_1 and D_2.*

If there is a resolution tree T of F with the root labeled C, we say C can be derived from F by resolution. A basic fact about resolution is that F logically implies C if and only if there is a clause $D \subseteq C$ that can be derived from F by resolution. Hence, F is unsatisfiable iff $\square$ can be derived from F by resolution.

Theorem 4.2. *Let F be an unsatisfiable linear k-CNF formula. Then any resolution tree of F whose root is labeled with $\square$ has at least $2^{2^{\frac{k-1}{2}}-1}$ nodes.*

We see that although there exist exponentially large unsatisfiable linear k-CNF formulas, every resolution tree proving their unsatisfiability must have *doubly exponential* size. Our second result in this section is about a rather strict notion of "direct constructions".

Definition 4.3. *A resolution tree T of F is called* strict *if every clause of F appears exactly once as a a label of a leaf of T.*

The difference is that in a non-strict resolution tree, several leaves can be labeled with the same clause of F. We call F *strictly treelike* if there is a strict resolution tree T of F whose root is labeled with $\square$.

Definition 4.4. *A formula F is* recursively decomposable *if either*

- *$F = \{\square\}$, or*
- *there is a variable x and a bipartition $F = F_0 \uplus F_1$, such that $F_0^{[x \mapsto 0]}$ and $F_1^{[x \mapsto 1]}$ are both recursively decomposable.*

Lemma 4.5. *If F is minimal unsatisfiable, then it is strictly treelike if and only if it is recursively decomposable.*

The proof uses induction over the structure of the strict resolution tree T and the tree implicitely defined in Definition 4.4 and is quite straightforward.

Let us give some examples. A smallest unsatisfiable k-CNF formula can be obtained by taking k variables $V = \{x_1, \dots x_k\}$, and setting F to be the set of all 2^k clauses over V. This is minimal unsatisfiable. We claim that F is strictly treelike. To see this, let $F_0 \subseteq F$ and $F_1 \subseteq F$ consist of all clauses containing x_k and $\bar{x}_k$, respectively. Then $F_0^{[x_k \mapsto 0]} = F_1^{[x_k \mapsto 1]}$, namely they consist of all clauses over $V \setminus \{x_k\}$, and by induction they

are strictly treelike (the base case is $k = 0$, where $F = \{\square\}$, of course). By Lemma 4.5, F is strictly treelike.

As a second example, take the formula of Theorem 3.1. Here, seeing that it is strictly treelike is easier from Definition 4.3 itself than from Definition 4.4.

As a third example, the formula constructed by Gebauer [8] that, up to small constant factor, achieves the lower bound of Kratochvíl, Savicky and Tuza [6] on d_k, is strictly treelike.

It should be said that our treelike formulas are closely related to so-called MU(1) formulas. A formula F is MU(1) if it is minimal unsatisfiable, and its number of clauses exceeds the number of variables by exactly 1. The value 1 is not chosen arbitrarily: It is the smallest possible value, since every minimal unsatisfiable formula has strictly more clauses than variables, as Aharoni and Linial [14] have shown. If in Definition 4.4, we require F_0 and F_1 to have no common variables besides x, we obtain exactly the class of MU(1) formulas. This has been shown by Davydov, Davydova and Kleine Büning [15].

MU(1) formulas have been used by Hoory and Szeider [16] to compute an approximation of $f(k)$, the maximum number d such that any k-CNF formula with F with $d(F) \leq d$ is satisfiable. Surprisingly, it is still unknown whether $f(k)$ is computable at all. However, if one defines its restriction to MU(1) formulas, say, define $f'(k)$ to be the maximum d such that there is no MU(1) formula F with $d(F) \leq d$, then $f'(k)$ is computable, and, as a consequence of the result of Gebauer [8], $f'(k)$ approximates $f(k)$ up to constant factor. Our result will be in stark contrast to this. We will prove that no strictly treelike linear k-CNF formula, hence neither any linear MU(1) formula, is significantly smaller than the formula of Theorem 3.1.

Theorem 4.6. *For any $\epsilon > 0$, there exists a constant c such that that any strictly treelike linear k-CNF formula has at least* $\mathrm{tower}_{2-\epsilon}(k - c)$ *clauses.*

4.1 Proofs

Proof (of Theorem 4). Let F be an unsatisfiable linear k-CNF formula, and let T be a resolution tree of F, whose root is labeled with $\square$. We want to show that T has a large number of nodes. It is not difficult to see that a resolution tree of minimal size is *regular*, meaning that no variable is resolved more than once on a path from the root to a leaf. See Urquhart [17], Lemma 5.1, for a proof of this fact. Hence we assume that T is a regular resolution tree. We take a random walk of length ℓ in T starting at the root, in every step choosing randomly to go to one of the two children of the current vertex. If we arrive at a leaf, we stay there. We claim that if $\ell \leq \sqrt{2^{k-1}}$, then with probability at least $\frac{1}{2}$, our walk does not end at a leaf. Hence, at least half of all possible 2^ℓ possible walks lead to distinct nodes, and thus T has at least $2^{2^{\frac{k-1}{2}}-1}$ nodes.

To analyze our random walk, note that each edge in T can be associated with an assignment to the resolved variable, as illustrated in Figure 4.1: If C is the resolvent of D_1 and D_2, x the resolved variable, and $x \in D_1$ and $\bar{x} \in D_2$, we label the edge from C to D_1 by $x \mapsto 0$ and from C to D_2 by $x \mapsto 1$. Every path from the root to a node gives a partial assignment, by simply combining all variable assignments on its edges. Since T is a regular resolution tree, no variable is assigned twice on that path.

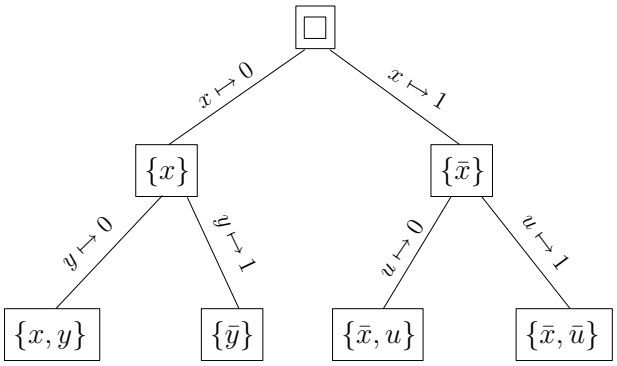

Fig. 1. A resolution tree, with its edge labeled in the obvious way. Every clause is unsatisfied when applying the assignments on the path to the root.

If p is a path in T leading from the root to a node labeled C, and α is the partial assignment associated with p, then $C^{[\alpha]} = \square$. In our random walk, let α_i denote the partial assignment associated with the first i steps. Hence α_0 is the empty assignment, and α_i assigns exactly i variables (if we are not yet at a leaf). Write $F_i := F^{[\alpha_i]}$. The α_i and F_i are random objects. For a formula G, we define the *weight* $w(G)$ to be

$$w(G) := \sum_{C \in G, |C| \leq k-2} 2^{k-|C|} \ .$$

Since F is a k-CNF formula, $w(F) = 0$, and if a formula G contains the empty clause, then $w(G) \geq 2^k$.

Lemma 4.7.

$$\mathbb{E}[w(F_{i+1})] \leq \mathbb{E}[w(F)_i] + 2i \ .$$

Since $w(F_0) = 0$, this immediately implies $\mathbb{E}[w(F_\ell)] \leq 2\binom{\ell}{2} \leq \ell^2$. If our random walk ends at a leaf labeled $C \in F$, then $\square = C^{[\alpha_\ell]} \in F_\ell$, and $w(F_\ell) \geq 2^k$. Therefore

$$\ell^2 \geq \mathbb{E}[w(F_\ell)] \geq 2^k \Pr[\text{ the random walk ends at a leaf }] \ .$$

We conclude that at least half of all paths of length $\ell^* = \sqrt{2^{k-1}}$ starting at the root do not lead to a leaf, and T has at least 2^{ℓ^*-1} internal nodes, which proves the theorem. It remains to prove the lemma.

Proof (of the lemma). For a formula G and a variable x, let $d_{k-1}(x, G)$ denote the number of $(k-1)$-clauses containing x or $\bar{x}$. Since $F = F_0$ is a k-CNF formula, $d_{k-1}(x, F) = 0$, for all variables x. We claim that for any x and any step i in our random walk, $d_{k-1}(x, F_i) \leq d_{k-1}(x, F_{i+1}) + 1$. Why is this so? If we are already at a leaf, we do not extend the partial assignment α_i, hence $d_{k-1}(x, F_i) = d_{k-1}(x, F_{i+1})$. Otherwise, F_{i+1} is obtained from F_i by setting some variable y to 0 or 1. If $x = y$, then $d_{k-1}(x, F_{i+1}) = 0$. There is only one possibility how $d_{k-1}(x, F_{i+1})$ can be larger than $d_{k-1}(x, F_i)$, namely that some k-clause C contains both x and y, and y is set such that C becomes a $(k-1)$-clause in F_{i+1}. Since F is linear, there is at most one such clause (this is the only point where we use linearity). We see that $d_{k-1}(x, F_{i+1}) \leq d_{k-1}(x, F_i) + 1$, and therefore $d_{k-1}(x, F_i) \leq i$. Consider $w(F_i)$, which was defined as

$$w(F_i) = \sum_{C \in F_i, |C| \leq k-2} 2^{k-|C|} \ .$$

Let F_{i+1} be obtained from F_i by setting y to 0 or 1. If $C \in F_i$ does not contain y nor $\bar{y}$, then C contributes as much to $w(F_{i+1})$ as to $w(F_i)$. If C contains y, and $|C| \leq k-2$, then with probability $\frac{1}{2}$, y is set such that it satisfies C, and its contribution to $w(F_{i+1})$ will be 0. Also with probability $\frac{1}{2}$, y is set such that C shrinks by one literal, so its contribution to $w(F_{i+1})$ doubles. On expectation, its contribution stays the same. If $|C| = k$, it does not contribute to $w(F_i)$ nor to $w(F_{i+1})$. The only case is if $|C| = k-1$ and it contains y. Then its contribution to $w(F_i)$ is 0, and with probability $\frac{1}{2}$, y is set such that C becomes a $(k-2)$-clause in F_{i+1} and its contribution to $w(F_{i+1})$ is 4. Hence on expectation each clause containing y increases its contribution by 2. By the above observation, there are at most $d_{k-1}(y, F_i) \leq i$ such clauses, hence $\mathbb{E}[w(F_{i+1})] \leq \mathbb{E}[w(F_i)] + 2i$. □

This concludes the proof of the theorem. □

4.2 Proof of Theorem 4.6

Recall that we are dealing with a strictly treelike linear k-CNF formula F. This means there is a resolution tree T of F whose root is labeled with $\square$ and the leaves of T are in 1–1-correspondence with the clauses of F. We want to show that the size of T is at least $\text{tower}_{2-\epsilon}(k-c)$. One would like to define a complexity measure for clauses appearing as labels of nodes in T that is huge for leaves and small for the root, and does not decrease too much in a resolution step. We actually do something slightly different, namely defining a whole vector of complexity measures for each clause. To say some words about notation, we will use letters u, v, w to denote nodes in the resolution tree, C, D to denote clauses, and x, y, z to denote variables.

For each node u in T, let C_u be the clause u is labeled with, and let F_u be set of clauses occurring as labels at the leaves of the subtree of T rooted at u. Hence $F_u \subseteq F$, and if u is a leaf, then $F_u = \{C_u\}$, and $F_{\text{root}} = F$. With every node u of T we associate a graph G_u with vertex set C_u, and two literals connected by an edge if they are both containined in some $D \in F_u$. Resolution now has a simple interpretation as a "calculus on graphs". See Figure 4.2. If u is a leaf, then $G_u = K_k$, and G_{root} is the empty graph $(\emptyset, \emptyset)$, containing no vertices.

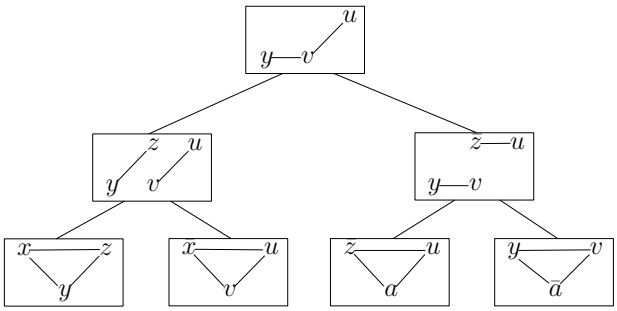

Fig. 2. Resolution as a calculus on graphs. A resolution step amounts to deleting the resolved vertex and taking the union of the two graphs. If F is linear and T is strictly treelike, then no union produces multiple edges.

We define our "complexity measure" for nodes u in T in terms of G_u. For a graph G, let $\kappa_i(G)$ denote the minimum size of a set $U \subseteq V(G)$ such that $G - U$ contains no

i-clique. Thus, $\kappa_1(G) = |V(G)|$, and $\kappa_2(G)$ is the size of a minimum vertex cover of G. For the complete graph K_k, we have $\kappa_i(K_k) = k - i + 1$. We write $\kappa_i(u) := \kappa_i(G_u)$.

Proposition 4.8. *If v is a child of u in T, then $\kappa_i(u) \geq \kappa_i(v) - 1$.*

Proof. The follows from the fact that removing one vertex can decrease κ_i by at most one, and adding vertices and edges can only increase it. Let C_u be the resolvent of C_v and C_w, and let x be the resolved literal. We assume w. l. o. g. that $x \in C_v$ and $\bar{x} \in C_w$. By the definition of resolution, $C_v - \{x\} \subseteq C_u$, therefore $G_v - \{x\}$ is a subgraph of G_u. If there is a set $U \subseteq V(G_u)$ such that $G_u - U$ does not have an i-clique, then surely $G_v - (U \cup \{x\})$ does not have an i-clique, either. Thus $\kappa_i(v) \leq \kappa_i(u) + 1$. □

If u is an ancestor of v in T, let $\mathrm{dist}(u, v)$ denote the number of edges in the T-path from u to v. Repeatedly applying Proposition 4.8 yields that $\kappa_i(u) \geq \kappa_i(v) - \mathrm{dist}(u, v)$.

To prove Theorem 4.6, we have to define some parameters. These definitions look ugly, but the precise values of the parameters are artefacts of the proof and are not really important. Let k be as in the proof, i.e. the size of the clauses of F. Fix some value $1 \leq \ell \leq k$ and define α_i and θ_i for $1 \leq i \leq l$ as follows:

$$\begin{aligned}
\alpha_\ell &:= 1 \\
\theta_\ell &:= \left\lfloor \frac{k - \ell + 1}{2} \right\rfloor - 1 \\
\alpha_i &:= \frac{\alpha_{i+1}\theta_{i+1} - 1}{\theta_i} \left\lfloor \frac{\theta_i}{\theta_{i+1}} \right\rfloor , \quad 1 \leq i < \ell \\
\theta_i &:= \left\lfloor \frac{2^{\alpha_{i+1}\theta_{i+1}-2}}{\theta_{i+1}} \right\rfloor - 1 , \quad 1 \leq i < \ell
\end{aligned}$$

For the right value of ℓ, θ_1 is a tower function in k. More precisely, for any $\epsilon > 0$, there exists a $c \in \mathbb{N}$ such that when choosing $\ell = k - c$, then $\theta_1 \geq \mathrm{tower}_{2-e}(k - c)$.

Theorem 4.9. *Let F be a recursively decomposable linear k-CNF formula. Then F has at least $2^{\alpha_1\theta_1}$ clauses.*

Proof. We call a node u in T *i-extendable* if $\kappa_j(u) \leq \theta_j$ for each $i \leq j \leq \ell$. If u is i-extendable, by definition it is also $(i+1)$-extendable. For $i = \ell + 1$, the condition is void, so every node $(\ell + 1)$-extendable. Let r denote the root of T. Since $\theta_i \geq 0$ and $\kappa_i(r) = 0$ for all i, r is clearly 1-extendable.

Definition 4.10. *A set W of descendants of u in T such that (i) no vertex in W is an ancestor of any other vertex in W and (ii) $\mathrm{dist}(u, w) \leq d$ for all $w \in W$ is called an* antichain of u at distance $\leq d$. *If furthermore every $w \in W$ is i-extendable, we call W an i-extendable antichain.*

Lemma 4.11. *Let $1 \leq i \leq \ell$, and let u be a node in T. If u is i-extendable, then there is an $(i+1)$-extendable antichain W of u at distance $\leq \theta_i$ such that $|W| = 2^{\alpha_i\theta_i}$.*

The lemma implies the theorem: The root r of T is 1-extendable, and since each $w \in W$ has at least one leaf in its subtree, T has at least $2^{\alpha_1\theta_1}$ leaves. Hence it remains to prove the lemma.

Proof. We use induction on $\ell - i$. For the base case $i = \ell$, we have $\kappa_\ell \leq \theta_\ell$, as u is ℓ-extendable. Since each leaf v of T has $\kappa_\ell(v) = k - \ell + 1 \geq 2\theta_\ell + 2$, Proposition 4.8 tells us that every leaf in the subtree of u has distance at least $\theta_\ell + 2$ from u. Since T is a

complete binary tree, there are 2^{θ_ℓ} descendants of u at distance exactly θ_ℓ from u. This is the desired antichain W of u. Since every node is $(\ell+1)$-extendable by definition, the base case holds.

For the step, let $i < \ell$, i.e., u is i-extendable.

Proposition 4.12. *Let v be a descendant of u with $\mathrm{dist}(u,v) \leq \theta_i$. If v is $(i+1)$-extendable, then there is an $(i+1)$-antichain W of v at distance $\leq \theta_{i+1}$ of size $2^{\alpha_{i+1}\theta_{i+1}-1}$.*

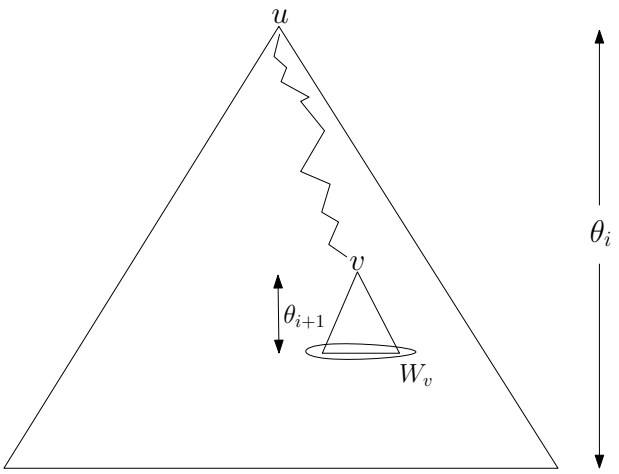

Fig. 3. Illustration of Proposition 4.12. If node u is i-extendable, and v is a not too far $(i+1)$-extendable descendant of u, then v itself has many not too far descendants W_v, at least half of which are $(i+1)$-extendable themselves.

Proof. By applying the induction hypothesis of the lemma to v, there is an $(i+2)$-extendable antichain W of v at distance $\leq \theta_{i+1}$ of size $2^{\alpha_{i+1}\theta_{i+1}}$. We will show that at least half the nodes $w \in W$ satisfy $\kappa_{i+1}(w) \leq \theta_{i+1}$, thus half of W is $(i+1)$-extendable. This will prove the proposition.

Assume for the sake of contradiction that $\kappa_{i+1}(w) \geq \theta_{i+1}+1$ for more than half the nodes $w \in W$. Let the set of these nodes be called W'. By assumption, $|W'| \geq 2^{\alpha_{i+1}\theta_{i+1}-1}$. Consider any $w \in W'$, and let P be the set of resolved variables on the path from w to v. Clearly, $|P| = \mathrm{dist}(v,w) \leq \theta_{i+1}$, and $\mathrm{vbl}(C_w) \setminus P \subseteq \mathrm{vbl}(C_v)$, simply by the way resolution works. Viewing P as a set of vertices, we see that $G_w - P$ is a subgraph of G_v. Since $|P| \leq \theta_{i+1}$ and $\kappa_{i+1}(w) \geq \theta_{i+1}+1$, $G_w - P$ contains at least one $(i+1)$-clique, by the definition of κ_{i+1}. This clique is also contained in G_v. Hence for each $w \in W'$, G_v contains at least one $(i+1)$-clique of G_w. These cliques need not be vertex disjoint. However, an edge $\{x,y\}$ in G_w for $w \in W$ means that there is a clause $C \in F$ occurring at a leaf in the subtree of w such that $x,y \in \mathrm{vbl}(C)$. Since F is linear, and W' is an antichain, it means that for two distinct $w_1, w_2 \in W'$, the graphs G_{w_1} and G_{w_2} do not share an edge. Hence each $w \in W'$ contributes an *edge-disjoint* copy of K_{i+1} to G_v. Therefore, G_v contains at least $|W'|$ edge-disjoint copies of K_{i+1}.

Since v is $(i+1)$-extendable, $\kappa_{i+1}(v) \leq \theta_{i+1}$, so there exists a set $U \subseteq V(G_v)$ of size $\leq \theta_{i+1}$ such that $G_v - U$ contains no $(i+1)$-clique. Each of the $|W'|$ edge-disjoint copies of K_{i+1} in G_v contains a vertex of U. Thus, there exists a vertex $x \in U$ contained in at least $\frac{|W'|}{|U|} \geq \frac{2^{\alpha_{i+1}\theta_{i+1}-1}}{\theta_{i+1}} \geq 2\theta_i + 1$ edge-disjoint copies of K_{i+1}. Two such copies cannot share any vertex besides x. Thus G_v contains at least $2\theta_i+1$ *vertex-disjoint* copies of K_i, and therefore $\kappa_i(v) \geq 2\theta+1$. By Proposition 4.8, $\kappa_i(u) \geq \kappa_i(v) - \mathrm{dist}(u,v) \geq \theta_i + 1$. This, however, contradicts the assumption of Lemma 4.11 that u is i-extendable. We conclude that $|W'| \leq \frac{1}{2}|W|$, which proves the proposition. □

Let us continue with the proof of the lemma. For some $(i+1)$-extendable descendant v of u such that $\mathrm{dist}(u,v) \leq \theta_i$, Proposition 4.12 tells us that v produces $2^{\alpha_{i+1}\theta_{i+1}-1}$ new $(i+1)$-extendable nodes w with $\mathrm{dist}(u,w) \leq \mathrm{dist}(u,v) + \theta_{i+1}$. Formally, we define a sequence $W_0, W_1, \ldots$ of $(i+1)$-extendable antichains for u such that $|W_{j+1}| \geq |W_j| 2^{\alpha_{i+1}\theta_{i+1}-1}$, and for every $w \in W_j$ we have $\mathrm{dist}(u,w) \leq j\theta_{i+1}$. We define $W_0 = \{u\}$. For given W_j with $(j+1)\theta_{i+1} \leq \theta_i$, we apply Proposition 4.12 to each $v \in W_j$, obtaining an antichain W_v whose vertices have distance $\leq (j+1)\theta_{i+1}$ from u. Set $W_{j+1} := \bigcup_{v \in W_j} W_v$. This is an $(i+1)$-extendable antichain for u at distance $j\theta_{i+1} + \theta_{i+1}$, and $|W_{j+1}| \geq |W_j| 2^{\alpha_{i+1}\theta_{i+1}-1}$. The W_j are defined for all $j \leq j^* := \left\lfloor \frac{\theta_i}{\theta_{i+1}} \right\rfloor$, and thus

$$|W_{j^*}| \geq \left(2^{\alpha_{i+1}\theta_{i+1}-1}\right)^{\left\lfloor \frac{\theta_i}{\theta_{i+1}} \right\rfloor} = 2^{\alpha_i \theta_i} .$$

Thus, W_{j^*} is the antichain of $(i+1)$-extendable nodes claimed by Lemma 4.11. This completes the proof of the lemma. □

As already argued, we can apply Lemma 4.11 to the root r of T, which is clearly 1-extendable, simply because $\kappa_i(r) = 0$ for all i, and obtain an antichain of size $2^{\alpha_1\theta_1}$ nodes. This proves the theorem. □

5 Open Problems

What is the asymptotic value of $m_{\mathrm{LIN}}(k)$? We have determined $m_{\mathrm{LIN}}(k)$ up to a polynomial factor in k. We do not have any strong feeling about whether the true value lies closer to the upper or to the lower bound.

How do unsatisfiable linear k-CNF formulas look like? We would really like to see an explicit construction of an unsatisfiable linear k-CNF formula of reasonable size. We suspect one has to find some algebraic construction.

What is the resolution complexity of linear k-CNF formulas? We have seen that the tree resolution complexity is doubly exponential in k. We suspect the same is true for general resolution.

References

1. Porschen, S., Speckenmeyer, E., Zhao, X.: Linear cnf formulas and satisfiability. Discrete Appl. Math. **157**(5) (2009) 1046–1068
2. Abbott, H.: An application of Ramsey's Theorem to a problem of Erdős and Hajnal. Canad. Math. Bull. (1965) 515–517
3. Hales, A.W., Jewett, R.I.: Regularity and Positional Games. Trans. Amer. Math. Soc. **106** (1963) 222–229
4. Kostochka, A., Mubayi, D., Rödl, V., Tetali, P.: On the chromatic number of set systems. Random Structures Algorithms **19**(2) (2001) 87–98
5. Tovey, C.A.: A simplified NP-complete satisfiability problem. Discrete Appl. Math. **8**(1) (1984) 85–89
6. Kratochvíl, J., Savický, P., Tuza, Z.: One more occurrence of variables makes satisfiability jump from trivial to NP-complete. SIAM Journal of Computing **22**(1) (1993) 203–210
7. Hoory, S., Szeider, S.: A note on unsatisfiable k-CNF formulas with few occurrences per variable. SIAM Journal on Discrete Mathematics **20**(2) (2006) 523–528
8. Gebauer, H.: Disproof of the neighborhood conjecture and its implications to sat (2008) submitted.
9. Scheder, D.: Unsatisfiable linear k-CNFs exist, for every k. CoRR **abs/0708.2336** (2007)
10. Erdős, P., Lovász, L.: Problems and results on 3-chromatic hypergraphs and some related questions. In Hajnal, A., Rado, R., Sós, V.T., eds.: Infinite and Finite Sets (to Paul Erdős on his 60th birthday), Vol. II. North-Holland (1975) 609–627
11. Buss, S., Pitassi, T.: Resolution and the weak pigeonhole principle. In: Computer Science Logic (Aarhus, 1997). Volume 1414 of Lecture Notes in Comput. Sci. Springer, Berlin (1998) 149–156
12. Tseitin, G.S.: On the complexity of derivations in the propositional calculus. Studies in Constructive Mathematics and Mathematical Logic (Part 2) (1968) 115–125
13. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow—resolution made simple. J. ACM **48**(2) (2001) 149–169
14. Aharoni, R., Linial, N.: Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas. J. Combin. Theory Ser. A **43**(2) (1986) 196–204
15. Davydov, G., Davydova, I., Büning, H.K.: An efficient algorithm for the minimal unsatisfiability problem for a subclass of CNF. Ann. Math. Artificial Intelligence **23**(3-4) (1998) 229–245
16. Hoory, S., Szeider, S.: Computing unsatisfiable k-SAT instances with few occurencesper variable. Theoretical Computer Science **337**(1-3) (2005) 347–359
17. Urquhart, A.: The complexity of propositional proofs. Bulletin of Symbolic Logic **1** (1995) 425–467