On the Satisfaction Probability of k-CNF Formulas

Till Tantau
Institute for Theoretical Computer Science, Universität zu Lübeck, Germany

Abstract

The satisfaction probability \( \sigma(\phi) := \Pr_{\beta \sim \text{vars}(\phi) \to \{0,1\}}[\beta \models \phi] \) of a propositional formula \( \phi \) is the likelihood that a random assignment \( \beta \) makes the formula true. We study the complexity of the problem \( k\text{SAT-PROB}_{>\delta} = \{ \phi \text{ is a } k\text{CNF formula \mid } \sigma(\phi) > \delta \} \) for fixed \( k \) and \( \delta \). While \( 3\text{SAT-PROB}_{>0} = 3\text{SAT} \) is NP-complete and \( SAT-\text{PROB}_{1/2} \) is PP-complete, Akmal and Williams recently showed \( 3\text{SAT-PROB}_{>1/2} \in P \) and \( 4\text{SAT-PROB}_{>1/2} \in \text{NP-complete} \); but the methods used to prove these striking results stay silent about, say, \( 4\text{SAT-PROB}_{>3/4} \), leaving the computational complexity of \( k\text{SAT-PROB}_{>\delta} \) open for most \( k \) and \( \delta \). In the present paper we give a complete characterization in the form of a trichotomy: \( k\text{SAT-PROB}_{>\delta} \) lies in \( \text{AC}^0 \), is \( \text{NL-complete} \), or is \( \text{NP-complete} \); and given \( k \) and \( \delta \) we can decide which of the three applies. The proof of the trichotomy hinges on a new order-theoretic insight: Every set of \( k\text{CNF} \) formulas contains a formula of maximal satisfaction probability. This deceptively simple result allows us to (1) kernelize \( k\text{SAT-PROB}_{>\delta} \), (2) show that the variables of the kernel form a strong backdoor set when the trichotomy states membership in \( \text{AC}^0 \) or \( \text{NL} \), and (3) prove a locality property by which for every \( k\text{CNF} \) formula \( \phi \) we have \( \sigma(\phi) \geq \delta \) iff \( \sigma(\psi) \geq \delta \) for every fixed-size subset \( \psi \) of \( \phi \)'s clauses. The locality property will allow us to prove a conjecture of Akmal and Williams: The majority-of-majority satisfaction problem for \( k\text{CNF} \) lies in \( \text{P} \) for all \( k \).

1 Introduction

For a propositional formula \( \phi \) like \( (x \lor y \lor z) \land (\neg x \lor \neg y) \land (y \lor z) \) it is, in general, a very hard problem to obtain much information about the number \( \#(\phi) \) of satisfying assignments or, equivalently, about the satisfaction probability \( \sigma(\phi) \) defined as

\[
\sigma(\phi) := \Pr_{\beta \sim \text{vars}(\phi) \to \{0,1\}}[\beta \models \phi] = \frac{\#(\phi)}{2^n}
\]

where \( n = |\text{vars}(\phi)| \) is the number of variables in \( \phi \). By Cook’s Theorem [7] it is already NP-complete to determine whether \( \sigma(\phi) > 0 \) holds; and to determine whether \( \sigma(\phi) > 1/2 \) holds is complete for PP. Indeed, the function \( \#(\cdot) \) itself is complete for \#P, a counting class high up in the complexity hierarchies. Writing \( SAT-\text{PROB}_{>\delta} \) for \( \{ \phi \mid \sigma(\phi) > \delta \} \), we can rephrase these results as “\( SAT-\text{PROB}_{>0} \) is NP-complete” (Cook’s Theorem) and “\( SAT-\text{PROB}_{1/2} \) is PP-complete” (and so is \( SAT-\text{PROB}_{>1/2} \), see for instance [13, Theorem 4.1]).

Cook’s result on the complexity of \( SAT-\text{PROB}_{>0} = SAT \) is quite robust regarding the kinds of formulas one considers: The problem stays NP-complete for formulas in CNFs, the set of formulas in conjunctive normal form, so \( CNF-\text{SAT-PROB}_{>0} = CNF-\text{SAT} \in \text{NP-complete} \), and even for formulas \( \phi \in 3\text{CNF} \), that is, when all clauses of \( \phi \) have at most three literals, so \( 3\text{SAT-PROB}_{>0} = 3\text{SAT} \in \text{NP-complete} \). Similarly, \( CNF-\text{SAT-PROB}_{>1/2} \) has the same complexity as \( SAT-\text{PROB}_{>1/2} \), see [13, lemma on page 80], and \( \#(\cdot) \) is still \#P-hard for formulas in \( 3\text{CNF} \) and even in \( 2\text{CNF} \), see [15].

© Till Tantau; licensed under Creative Commons License CC-BY 4.0
37th Computational Complexity Conference (CCC 2022).
Editor: Shachar Lovett; Article No. 2; pp. 2:1–2:27
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatics, Dagstuhl Publishing, Germany
On the Satisfaction Probability of \( k \)-CNF Formulas

In sharp contrast to these well-established hardness results, Akmal and Williams [2] recently showed that 3SAT-\( \text{PROB}_{>1/2} \) can be solved in polynomial time – in fact, they show this time bound for 3SAT-\( \text{PROB}_{>\delta} = \{ \phi \in 3\text{CNFS} \mid \sigma(\phi) > \delta \} \) for all rational \( \delta > 0 \) (intriguingly, their argument does not apply to non-rational \( \delta \) and leaves open the complexity of, say, 3SAT-\( \text{PROB}_{>1/\pi} \)). Yet again in contrast, they also show that 4SAT-\( \text{PROB}_{>1/2} \) is NP-complete for every \( k \geq 4 \). To complicate things even further, the NP-completeness result for 4SAT-\( \text{PROB}_{>1/2} \) can easily be extended to 4SAT-\( \text{PROB}_{>1/4} \), to 4SAT-\( \text{PROB}_{>1/8} \), to 4SAT-\( \text{PROB}_{>1/16} \) and so on, and with very little extra work to more exotic values of \( \delta \) like \( \delta = 15/32 \) – but apparently not to certain other values like \( \delta = 3/4 \) or \( \delta = 63/128 \). Indeed, 4SAT-\( \text{PROB}_{>15/16} \) is a trivial problem (lies in \( \text{AC}^0 \)) as every nonempty formula in 4CNFS has a satisfaction probability of at most 15/16 (a single nontrivial clause already rules out 1/16th of all assignments). In other words, even for a fixed \( k \) the complexity of 4SAT-\( \text{PROB}_{>\delta} \) might fluctuate wildly for changing \( \delta \) (does, in fact, as Figure 1 makes quite clear) and it is unclear how the methods introduced in [2] could be used to show that, say, 4SAT-\( \text{PROB}_{>63/128} \) is NL-complete while 3SAT-\( \text{PROB}_{>1/\pi} \) lies in \( \text{AC}^0 \).

1.1 Contributions of This Paper

In the present paper we continue the investigation of the complexity of the satisfaction probability function for \( k\text{CNFS} \) formulas initiated by Akmal and Williams. We will look at this complexity from three different angles – order-theoretic, algorithmic, and complexity-theoretic – and now sketch the main results we obtain for each of these aspects. (A small remark on notation first, however: For convenience, in this paper we make no difference between a CNF formula and its set of clauses. Each \( \phi \in \text{CNFS} \) is a finite set of clauses, which are sets of literals – so the formula from the paper’s first line is actually \( \phi = \{ x, y, z \}, \{ \neg x, \neg y \}, \{ y, z \} \} \in 3\text{CNFS} \) – and we forbid already syntactically that clauses contain both a variable and its negation, so \( \{ x, \neg x \} \notin \text{CNFS} \), while \( \{ x \}, \{ \neg x \} \} \in 1\text{CNFS} \).)
First, we address the order-theoretic properties of the number set $\text{kCNFS-}\sigma\text{-SPECTRUM} := \{\sigma(\phi) \mid \phi \in \text{CNFS}\} \subseteq [0, 1]$ for different $k$ and show that for all $k$ the following holds:

**Theorem 1.1** (Spectral Well-Ordering Theorem). $\text{kCNFS-}\sigma\text{-SPECTRUM}$ is well-ordered by $>$. Here, a set $X \subseteq [0, 1]$ is called well-ordered by $>$ if there is no infinite strictly increasing sequence of elements of $X$ or, equivalently, if every subset of $X$ contains a maximal element.

An equivalent way of stating the Spectral Well-Ordering Theorem is as follows (and observe that the corollary certainly does not hold when we replace “maximal” by “minimal” as the set $\Phi = \{\{x_1\}, \ldots, \{x_n\}\} \mid n \in \mathbb{N}\}$ shows):

**Corollary 1.2.** Every $\Phi \subseteq \text{CNFS}$ contains a formula of maximal satisfaction probability.

Yet another equivalent way of stating the Spectral Well-Ordering Theorem is terms of the spectral gaps below $\delta \in [0, 1]$:

**Definition 1.3.** Let $\text{spectral-gap}_{\text{CNFS}}(\delta) := \sup\{\epsilon \mid (\delta - \epsilon, \delta) \cap \text{CNFS-}\sigma\text{-SPECTRUM} = \emptyset\}$. In Figure 1, the spectral gaps can be “seen” directly: For any position $\delta$ (not necessarily a member of the spectrum), the spectral gap “stretches left till the next cross (member of the spectrum).” The key observation is that there always is a stretch:

**Corollary 1.4.** For all $k$ and $\delta \in [0, 1]$, we have $\text{spectral-gap}_{\text{CNFS}}(\delta) > 0$. As we will see, this deceptively simple statement has far-reaching consequences.

Second, we use the existence of spectral gaps below all $\delta \in [0, 1]$ to develop three new algorithmic methods for showing $\text{ksat-prob}_{\geq \delta} \in \text{AC}^0$ (note the “$\geq \delta$” rather than “$\geq \delta$” subscript). They are simpler than the algorithm of Akmal and Williams and make the tools of fixed-parameter tractability (FPT) theory accessible for the analysis. One of them, while impractical and having by far the worst resource bounds of the three algorithms (but still in P), has an underlying idea that is of independent interest (recall that we consider CNF formulas to be sets of clauses):

**Lemma 1.5** (Threshold Locality Lemma). For each $k$ and each $\delta \in [0, 1]$ there is a $C \in \mathbb{N}$ so that for all $\phi \in \text{CNFS}$ we have $\sigma(\phi) \geq \delta$, iff $\sigma(\psi) \geq \delta$ holds for all $\psi \subseteq \phi$ with $|\psi| \leq C$.

While the above lemma is foremost a mathematical statement about properties of $\text{CNF}$ formulas, we can trivially derive a decision algorithm for $\text{ksat-prob}_{\geq \delta}$ from it (Algorithm 2 later on): Iterate over all $\psi \subseteq \phi$ with $|\psi| \leq C$ and compute $\sigma(\psi)$ by brute force ($|\psi|$ is constant) and accept if all computed values are at least $\delta$.

The Threshold Locality Lemma also lies at the heart of a proof of $\text{maj-maj-ksat} \in \text{AC}^0$ for all $k$. The problem is defined as follows: Given a formula $\phi \in \text{CNFS}$ with $\text{vars}(\phi) \subseteq X \cup Y$, determine whether for a majority of the assignments $\beta : X \rightarrow \{0, 1\}$ the majority of extensions of $\beta$ to $\beta' : X \cup Y \rightarrow \{0, 1\}$ makes $\phi$ true. Akmal and Williams [2] conjectured $\text{maj-maj-ksat} \in \text{P}$. Corollary 3.10 shows that this is, indeed, the case.

Third, we apply the developed theory to $\text{ksat-prob}_{\geq \delta}$: A problem whose complexity is somewhat more, well, complex than that of $\text{ksat-prob}_{\geq \delta}$. It turns out that a complete classification of the complexity for all values of $k$ and $\delta$ is possible in the form of a trichotomy:

**Theorem 1.6** (Spectral Trichotomy Theorem). Let $k \geq 1$ and $\delta \in [0, 1]$ be a real number. Then $\text{ksat-prob}_{\geq \delta}$ is $\text{NP}$-complete or $\text{NL}$-complete or lies in $\text{AC}^0$.

In the following, we explore each of the above “points of view” in a bit more detail and have a brief look at the main ideas behind the proofs of the main results. Later, each angle will be addressed in detail in a dedicated main section of this paper.
Order-Theoretic Results. In a sense, the “reason” why the functions $\sigma(\cdot)$ and $\#(\cdot)$ are so hard to compute in general, lies in the fact that the spectrum $\CNFS-\sigma\text{-SPECTRUM} = \{ \sigma(\phi) \mid \phi \in \CNFS \}$ is just the set $\mathbb{D}$ of dyadic rationals (numbers of the form $m/2^e$ for integers $m$ and $e$) between 0 and 1 (see Lemma 2.1). In particular, it is a dense subset of $[0,1]$ and in order to conclusively decide whether, say, $\sigma(\phi) \geq 1/3$ holds for an arbitrary $\phi \in \CNFS$, we may need to determine all of the first $n$ bits of $\sigma(\phi)$.

A key insight of Akmal and Williams is that for fixed $k$, the spectra $k\CNFS-\sigma\text{-SPECTRUM}$ behave differently, at least near to 1: There are “holes” like $3\CNFS-\sigma\text{-SPECTRUM} \cap (7/8, 1) = \emptyset$ since for a 3CNF formula $\phi$ we cannot have $7/8 < \sigma(\phi) < 1$ (a single size-3 clause already lowers the satisfaction probability to at most $7/8$). This implies immediately that, say, $3\SAT-\text{PROB}_{\geq 9/10}$ is actually a quite trivial problem: The only formula in $3\CNFS$ having a satisfaction probability larger than $9/10$ has probability 1 and is the trivial-to-detected tautology $\phi = \emptyset$. In general, for all $k$ we have $k\CNFS-\sigma\text{-SPECTRUM} \cap (1 - 2^{-k}, 1) = \emptyset$.

Of course, $3\CNFS-\sigma\text{-SPECTRUM}$ does not have “holes above every number $\delta$” as we can get arbitrarily close to, say, $\delta = 3/4$: Just consider the sequence of 3CNF formulas $\phi_1 = \{\{a, b, x_1\}\}$, $\phi_2 = \{\{a, b, x_1\}, \{a, b, x_2\}\}$, $\phi_3 = \{\{a, b, x_1\}, \{a, b, x_2\}, \{a, b, x_3\}\}$, and so on with $\sigma(\phi_i) = 3/4 + 2^{-i-2}$ and $\lim_{i \to \infty} \sigma(\phi_i) = 3/4$. Nevertheless, Akmal and Williams point out that their algorithm is in some sense based on the intuition that there are “lots of holes” in $k\CNFS-\sigma\text{-SPECTRUM}$. The new Spectral Well-Ordering Theorem, Theorem 1.1 above, turns this intuition into a formal statement.

Well-orderings are a standard notion of order theory; we will just need the special case that we are given a set $X$ of non-negative reals and consider the total order $> \cap$ on it. Then $X$ is well-ordered (by $>$) if there is no infinite strictly increasing sequence $x_0 < x_1 < x_2 < \cdots$ of numbers $x_i \in X$ or, equivalently, if $X$ is bounded and for every $x \in \mathbb{R}_{\geq 0}$ there is an $\epsilon > 0$ such that $(x - \epsilon, x) \cap X = \emptyset$ or, again equivalently, if every subset of $X$ contains a maximal element. In particular, Theorem 1.1 tells us that every $\Phi \subseteq k\CNFS$ contains a formula $\phi \in \Phi$ of maximal satisfaction probability, that is, $\sigma(\phi) \geq \sigma(\phi')$ for all $\phi' \in \Phi$.

Observe that $k\CNFS-\sigma\text{-SPECTRUM}$ is certainly not well-ordered by $\lt$, only by $\gt$. However, with a small amount of additional work we will be able to show that it is at least topologically closed, see Lemma 3.5. A succinct way of stating both this and Theorem 1.1 is that for every set $X \subseteq k\CNFS-\sigma\text{-SPECTRUM}$ of numbers we have $\sup X \in X$ and $\inf X \in k\CNFS-\sigma\text{-SPECTRUM}$. An interesting corollary of this is that non-rational, non-dyadic thresholds can always be replaced by dyadic rationals (recall $\mathbb{D} = \{m/2^e \mid m, e \in \mathbb{Z}\}$):

**Corollary 1.7 (of Lemma 3.5).** For every non-dyadic $\delta \in [0, 1] \setminus \mathbb{D}$, for $\delta' = \inf\{\sigma(\phi) \mid \phi \in \CNFS, \sigma(\phi) > \delta\} \in \mathbb{D}$ we have $k\SAT-\text{PROB}_{\geq \delta} = k\SAT-\text{PROB}_{\geq \delta'}$.

The proof of Theorem 1.1 will need only basic properties of well-ordered sets of reals, like their being closed under finite sums and unions, and the following simple relationship (which also underlies the analysis of Akmal and Williams [2]) between the satisfaction probability of a formula $\phi$ and the size of packings $\pi \subseteq \phi$, which are just sets of pairwise variable-disjoint clauses:

**Lemma 1.8 (Packing Probability Lemma).** Let $\phi \in \CNFS$ and let $\pi \subseteq \phi$ be a packing. Then $\sigma(\phi) \leq (1 - 2^{-k})|\pi| \gt$ and, equivalently, $\log_{1 - 2^{-k}}(\sigma(\phi)) \geq |\pi|$.

**Proof.** We have $\sigma(\phi) \leq \sigma(\pi) = \prod_{i \in \pi} (1 - 2^{-|\pi'|}) \leq (1 - 2^{-k})|\pi|$ as all clauses of $\pi$ are variable-disjoint and, hence, their satisfaction probabilities are pairwise independent. □

A simple consequence of the Packing Probability Lemma will be that for every $\phi \in k\CNFS$, we can write $\sigma(\phi)$ as a sum $\sum_{i=1}^{s} \sigma(\phi_i)$ with $\phi_i \in (k - 1)\CNFS$ in such a way that $s$ depends only on $\sigma(\phi)$, which will almost immediately yield Theorem 1.1.
Algorithmic Results. The algorithm for deciding $k\text{SAT-PROB}_{\geq \delta}$ in [2] is complex (in the words of the authors from the technical report version [1]: “it depends on quite a few parameters, so the analysis becomes rather technical” and, indeed, the description of these parameters and constants alone takes one and a half pages, followed by eleven pages of analysis). Surprisingly, the purely order-theoretic Theorem 1.1 allows us to derive three simple algorithms for $k\text{SAT-PROB}_{\geq \delta}$, Algorithms 1, 2 and 3, whose underlying ideas are briefly described in the following.

![Figure 2](image)

**Figure 2** On the left, a formula $\phi \in 5\text{CNFS}$ is visualized by drawing, for each clause in $\phi$, a line that “touches” exactly the clause’s literals; so the upper dashed line represents the clause $\{f, x, y, z\}$. The “solid clauses” (meaning “clauses represented by a solid line”) form a sunflower $\psi \subseteq \phi$ with core $c = \{x, \neg y, z\}$. All red clauses, including the dotted clauses, are part of the link of this core, but the dotted clauses are not part of the sunflower: The upper dotted clause $\{x, \neg y, z, l, \neg e\}$ shares the literal “$l$” with the petal $\{x, \neg y, z, l, m\}$ of the sunflower, while the second dotted clause shares the variable “$l$” (though not the literal) with this petal. The dashed clauses are not part of the link (let alone the sunflower) as they do not contain all of the literals of the core (containing the variables is not enough). A key property of a sunflower is that it is “unlikely that an assignment makes the sunflower true, but not its core”: For $\phi$, this happens only when $g, j$, and $\neg k$ are all set to true as well as at least one of $\neg h$ or $i$, and one of $l$ or $m$. The probability that all of this happens is just $\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{3}{4} = \frac{3}{16}$, meaning that this is the maximal difference in the satisfaction probabilities of $\phi$ and the formula shown right, the result of “collapsing the link to the core $c$”.

The first algorithm is based on sunflowers, just like that of Akmal and Williams, but uses them in (far) less complex ways. They can be thought of as generalizations of packings, which are, indeed, the special case of a sunflower with an empty core.

**Definition 1.9.** A sunflower with core $c$ is a formula $\psi \in \text{CNFS}$ such that $c \subseteq e$ holds for all $e \in \psi$ and such that for any two different $e, e' \in \psi$ we have $\text{vars}(e) \cap \text{vars}(e') = \text{vars}(c)$.

In other words, the clauses of a sunflower “agree on the literals in $c$, but are variable-disjoint otherwise”, see the clauses represented by solid lines in Figure 2 for an example.

The importance of sunflowers lies in an easy observation: If $\psi$ is a sunflower with core $c$, then $\sigma(\psi) = \sigma(\{c\}) + \sigma(\{e \setminus c \mid e \in \psi\})$. As $\{e \setminus c \mid e \in \psi\}$ is clearly a packing, the Packing Probability Lemma implies that the probability that an assignment $\beta$ satisfies a sunflower $\psi$, but not its core, is at most $(1 - 2^{-k})^{\vert \psi \vert}$, which is a value that decreases exponentially as the size $\vert \psi \vert$ of the sunflower increases. Now for fixed $k$ and $\delta \in [0, 1]$ consider Algorithm 1. Function KERNELIZE($\phi$) will be familiar to readers interested in FPT theory: This is exactly the standard kernel algorithm for the hitting set problem with bounded hyperedge size based on “collapsing sunflowers” or, more precisely, on “collapsing the links $\text{link}_\phi(c) = \{e \in \phi \mid c \subseteq e\}$ for cores $c$ of large sunflowers.” In particular, standard results from
Algorithm 1 The sunflower-collapsing algorithm decides whether \( \sigma(\phi) \geq \delta \) holds for \( \phi \in \text{kCNFS} \) and fixed \( \delta \in [0,1] \). The "spectral gap" in the algorithm is the size of the "hole" below \( \delta \) in \text{kCNFS-\( \sigma \)-spectrum}, see Definition 1.3. The \( \text{link}_\phi(c) \) is the simply the set \( \{e \in \phi \mid c \subseteq e\} \). The test \( \sigma(\kappa) \geq \delta \) can be performed by "brute force" as the returned kernel \( \kappa \) will have fixed size. The algorithm's correctness follows from the fact that in a collapsing step (the assignment in line 3), the satisfaction probability cannot "tunnel through" the spectral gap below \( \delta \), see Figure 3.

1. **Algorithm** \text{KERNELIZE}(\phi, h)
   
   ```
   \text{while } \phi \text{ contains a sunflower of size at least } h + 1 \text{ with some core } c \text{ do}
   \phi \leftarrow (\phi \setminus \text{link}_\phi(c)) \cup \{c\}
   \text{return } \phi
   ```

2. **Algorithm** \text{SUNFLOWER-COLLAPSING}(\phi) // \( \phi \in \text{kCNFS} \) must hold
   
   ```
   \kappa \leftarrow \text{KERNELIZE}(\phi, \log_{1+2^{-k}}(\text{spectral-gap}_{\text{kCNFS}}(\delta))) // \text{See Definition 1.3}
   \text{if } \sigma(\kappa) \geq \delta \text{ then return } "\sigma(\phi) \geq \delta" \text{ else return } "\sigma(\phi) < \delta"
   ```

FPT theory (in this case, Erdős' Sunflower Lemma) tell us that \( \kappa \) will have a size depending only on \( h \) and \( k \), and, thus, \( \kappa \) has constant size (that is, a size depending only on \( h \)) and we can clearly compute \( \sigma(\kappa) \) in constant time when \( \kappa \) has only a constant number of clauses and hence also a constant number of variables.

The crucial question is, of course, why this algorithm works, that is, why should \( \sigma(\kappa) \geq \delta \) iff \( \sigma(\phi) \geq \delta \) hold? The deeper reason is Corollary 1.4 to the Spectral Well-Ordering Theorem, by which there is a gap of size \( \epsilon := \text{spectral-gap}_{\text{kCNFS}}(\delta) > 0 \) below \( \delta \) in \text{kCNFS-\( \sigma \)-spectrum} and we cannot tunnel through this gap by the following lemma (see also Figure 3):

![Figure 3](image)

**Figure 3** Three ways how \( \sigma(\phi) \) can change due to the assignment \( \phi' \leftarrow (\phi \setminus \text{link}_\phi(c)) \cup \{c\} \) in Algorithm 1: In each row, the crosses in the red lines are elements of \text{kCNFS-\( \sigma \)-spectrum} smaller than \( \delta \), while the crosses on the green lines are at least \( \delta \). In the first line, \( \sigma(\phi) \geq \delta \) holds and each assignment yields a new \( \phi' \) such that \( \sigma(\phi') \) is not larger – but \( |\sigma(\phi) - \sigma(\phi')| < \text{spectral-gap}_{\text{kCNFS}}(\delta) \) ensures that \( \sigma(\phi'') \) gets "stuck" at \( \delta \) as it cannot "tunnel through" the spectral gap by Lemma 1.10 and the dashed arrow is an impossible change in the satisfaction probability. In the second line, \( \sigma(\phi) \) is stuck at a much larger value than \( \delta \). In the third line, \( \sigma(\phi) \) is below \( \delta \) and can get smaller and smaller (unless there is another gap of size \text{spectral-gap}_{\text{kCNFS}}(\delta) further down).

**Lemma 1.10 (No Tunneling Lemma).** For \( \delta \in [0,1] \), let any two \( \phi, \phi' \in \text{kCNFS} \) be given with \( |\sigma(\phi) - \sigma(\phi')| < \text{spectral-gap}_{\text{kCNFS}}(\delta) \). Then \( \sigma(\phi) \geq \delta \) iff \( \sigma(\phi') \geq \delta \).

**Proof.** W.L.o.g. \( \sigma(\phi) \geq \sigma(\phi') \). Clearly \( \sigma(\phi') \geq \delta \) implies \( \sigma(\phi) \geq \delta \). If \( \sigma(\phi) \geq \delta \), then \( \sigma(\phi') > \delta - \text{spectral-gap}_{\text{kCNFS}}(\delta) \). As \( \sigma(\phi') \) cannot lie in the spectral gap, \( \sigma(\phi') \geq \delta \).
The crucial assignment $\phi' = (\phi \setminus \text{link}_\phi(e)) \cup \{e\}$ in the kernelization removes the link of $c$ in $\phi$, which is just the set of all superclauses of $c$ (which includes all clauses of the sunflower), but adds the core. This can only reduce the satisfaction probability of $\phi$ by at most $\sigma(\{e \setminus c \mid e \in \psi\})$. As this probability is at most $(1 - 2^{-k})^{\psi}$ by the Packing Probability Lemma, if the sunflower is larger than $h = \log_{1 - 2^{-k}}(\text{spectral-gap}_{\text{CNFs}}(\delta))$, then

$\sigma(\phi) - \sigma(\phi') < \text{spectral-gap}_{\text{CNFs}}(\delta)$. Thus, by the No Tunneling Lemma, $\sigma(\phi) \geq \delta$ iff $\sigma(\phi') \geq \delta$ as it is impossible that the satisfaction probability of $\phi$ “tunnels through the interval $(\delta - \text{spectral-gap}_{\text{CNFs}}(\delta), \delta)$” in any step of the while loop.

Algorithm 2

A simple algorithm for deciding $\text{ksat-prob} \geq \delta$. Note that while the algorithm runs in polynomial time (in fact, in $\text{AC}^0$), on input $\phi$ the runtime is of the form $(2|\phi|)^{C+1}$ for a (possibly huge) constant $C$. Phrased in terms of FPT theory, unlike Algorithm 1, the algorithm has an “XP-runtime” rather than an “FPT-runtime”.

```
algorithm locality-based($\phi$) // $\phi \in \text{cnfs}$ must hold
C ← $(2 + \log_{1 - 2^{-k}}(\text{spectral-gap}_{\text{CNFs}}(\delta)))^k \cdot k!
forall $\psi \subseteq \phi$ of size at most $C$ do // at most $|\phi|^{C+1}$ subsets
if $\sigma(\psi) < \delta$ then // check by brute force
  return "$\sigma(\psi) < \delta"$
return "$\sigma(\phi) \geq \delta"
```

Algorithm 2 is even simpler than the just-presented kernel algorithm. For its correctness, first note that the output in line 5 is certainly correct as $\psi \subseteq \phi$ implies $\sigma(\psi) \leq \sigma(\phi)$. To see that the output in line 6 is correct (which is essentially the claim of Lemma 1.5, the Threshold Locality Lemma), consider a $\phi$ for which $\sigma(\psi) < \delta$ holds. Starting with $\psi = \phi$, as long as possible, pick a clause $e \in \psi$ such that $\sigma(\psi \setminus \{e\}) < \delta$ still holds and set $\psi \leftarrow \psi \setminus \{e\}$. For the $\psi$ obtained in this way, $\sigma(\psi) < \delta$, $\psi \subseteq \phi$, and for all clauses $e \in \psi$ we have $\sigma(\psi \setminus \{e\}) \geq \delta$. Now, $\psi$ cannot contain a large sunflower since, otherwise, we could remove any petal $e$ of the sunflower and this would raise the satisfaction probability by at most the spectral gap. In particular, by the No Tunneling Lemma, $\sigma(\psi \setminus \{e\}) < \delta$ would still hold, contradicting that we can no longer remove clauses from $\psi$. By Erdős’ Sunflower Lemma we conclude that $|\psi| \leq C$ for a constant $C$ depending only on $k$ and $\delta$.

The Threshold Locality Lemma will also play a key role in the proof of Corollary 3.10, which states that $\text{maj-maj-ksat} \in \text{AC}^0$ holds for all $k$. In the proof, we use the lemma to turn certain satisfaction probability threshold problems (questions of the form “$\sigma(\rho) \geq \delta$?”) into model checking problems (questions of the form “$\beta = \omega$?”). This will allow us to present a reduction of $\text{maj-maj-ksat}$ to $\text{ksat-prob}_{\geq 1/2}$ for some (large) $l$ and we know already that the latter problem lies in $\text{AC}^0$.

The third algorithm addresses the well-established observation from FPT theory that while a kernel algorithm for a parameterized problem is in some sense the best one can hope for from a theoretical point of view, from a practical point of view it is also of high interest how we can actually decide whether $\sigma(\kappa) \geq \delta$ holds for a kernel: Of course, as the kernel size is fixed, this can be decided in constant time by brute-forcing all assignments – but a practical algorithm will need to use different ideas, such as those used in Algorithm 3. This algorithm uses the fact that if on input $\phi$ we can compute an interval $I \subseteq [0, 1]$ with $\sigma(\phi) \in I$ and $|I| < \epsilon = \text{spectral-gap}_{\text{CNFs}}(\delta)$, then we will have $\sigma(\phi) < \delta$ if $\max I < \delta$. The key insight is that we can compute such an interval recursively: If the interval returned by $\text{interval}(\phi)$ is not yet small enough, by the Packing Probability Lemma, $\phi$ must contain a small maximal packing, which we call $\text{pack}(\phi)$ and which can be obtained greedily. We can
Algorithm 3 A recursive algorithm for deciding $\sigma(\phi) \geq \delta$. The recursion computes an interval $I$ with $\sigma(\phi) \in I$ and $|I| \leq \epsilon$, by first obtaining a candidate interval using INTERVAL($\phi$). If this is too large, by the Packing Probability Lemma PACK($\phi$) (a maximal packing obtained greedily) is small and we can write $\sigma(\phi)$ as the sum $\sum_{\beta} \sigma(\phi|_{\beta})$, where $\phi|_{\beta}$ has as its models (= satisfying assignments) exactly all models $\beta'$ of $\phi$ that extend (or “agree with”) $\beta$. Crucially, $\phi|_{\beta}$ will be a $(k - 1)$-CNF formula, if $\phi$ was a $k$CNF formula, meaning that the recursion stops after at most $k$ steps. The addition $I + J$ of two intervals is defined as $\{i + j \mid i \in I, j \in J\}$.

```
algorithm PACK($\phi$) // computes a maximal packing $\pi \subseteq \phi$
    $\pi \leftarrow \emptyset$; foreach $c \in \phi$ do if vars($c$) \cap vars($\pi$) = $\emptyset$ then $\pi \leftarrow \pi \cup \{c\}$
    return $\pi$

algorithm INTERVAL($\phi$) // computes a possibly large interval $I$ with $\sigma(\phi) \in I$
    if PACK($\phi$) = $\phi$ then
        return $[\prod_{c \in \phi}(1 - 2^{-|c|}), \prod_{c \in \phi}(1 - 2^{-|c|})]$ // = [$\sigma(\phi), \sigma(\phi)$]
    else
        return $[0, (1 - 2^{-\kappa})|}\pi|]$

algorithm BOUNDED-INTERVAL($\phi, \epsilon$) // computes an interval $I$ with $\sigma(\phi) \in I$ and $|I| < \epsilon$
    if |INTERVAL($\phi$)| < $\epsilon$ then
        return INTERVAL($\phi$)
    else // $|\text{vars}(\text{PACK}(\phi))| \leq k \cdot |\text{PACK}(\phi)| \leq k \log_{1-2^{-\kappa}} \epsilon$
        return $\sum_{\beta: \text{vars}(\text{PACK}(\phi)) \rightarrow (0,1)} \text{BOUNDED-INTERVAL}(\phi|_{\beta}, \epsilon/2^{\text{vars}(\text{PACK}(\phi))})$

algorithm INTERVAL-BINDING($\phi$) // $\phi \in \text{CNFS}$ must hold
    $I \leftarrow \text{BOUNDED-INTERVAL}(\phi, \text{spectral-gap}_{\text{CNFS}}(\delta))$
    if $\max I < \delta$ then return $"\sigma(\phi) < \delta"$ else return $"\sigma(\phi) \geq \delta"$
```

then expand $\phi$ using this packing (also yet another key insight of Akmal and Williams [2]):

For a formula $\phi$ and an assignment $\beta: V \rightarrow \{0, 1\}$, where $V \neq \text{vars}(\phi)$ is permissible, we write $\phi|_{\beta}$ for the formula where we remove all clauses from $\phi$ that contain literals set to true by $\beta$, remove all literals from the remaining clauses that are set to false by $\beta$, and add one singleton clause for each literal set to true by $\beta$. For instance, for $\phi = \{a, b\}$, $\{\neg b, c, \neg f\}$, $\{d, e, g\}$ and $\beta: \{b, d\} \rightarrow \{0, 1\}$ with $\beta(b) = 1$ and $\beta(d) = 0$ we have $\phi|_{\beta} = \{\neg a, b\}$, $\{\neg d, c, \neg f\}, \{d, e, g\}\} \cup \{\{b\}, \{\neg d\}\} = \{\{c, \neg f\}, \{e, g\}, \{b\}, \{\neg d\}\}$. For a maximal packing $\pi \subseteq \phi$ we then have $\sigma(\phi) = \sum_{\beta: \text{vars}(\pi) \rightarrow (0,1)} \sigma(\phi|_{\beta})$ and, importantly, all $\phi|_{\beta}$ have a smaller maximum clause size. This will imply that Algorithm 3 correctly solves $k$SAT-PROB$_{\geq \delta}$.

Complexity-Theoretic Results. Our proof of Theorem 1.6, which states that $k$SAT-PROB$_{\geq \delta}$ is always NP-complete, NL-complete, or lies in AC$^0$, will be constructive as we can explicitly state for which values of $k$ and $\delta$ which case applies. For this statement, let us call $\delta \in [0, 1]$ a $k$-target for LCNFs if there is a formula $\omega \in \text{CNFS}$ with

$$\sigma(\omega) = \delta \quad \text{and} \quad \omega_{\geq k-1} \not\models \omega_{< k-1},$$

meaning that there is an assignment $\beta: \text{vars}(\omega) \rightarrow \{0, 1\}$ that is a model (= satisfying assignment) of $\omega_{\geq k-1} := \{c \in \omega \mid |c| > k - l\}$, but not of $\omega_{< k-1} := \{c \in \omega \mid |c| \leq k - l\}$ (“the large clauses do not imply the small clauses”). For instance, $\delta = 7/32$ is a 3-target for 2CNF as demonstrated by $\omega = \{a, b\}, \{c_1, c_2, c_3\} \in \text{3CNF}$ since

$$\sigma(\omega) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{7}{8} = \frac{7}{32} \quad \text{and} \quad \{\{c_1, c_2, c_3\}, \{a, b\}\} \not\models \{\{a, b\\}.$$

In contrast, no $\delta > 1/2$ can be a 3-target for 2CNFs as $\omega_{\geq 1} \not\models \omega_{\leq 1}$ implies that $\omega_{< 1}$ is not empty and, thus, $\omega \geq \omega_{< 1}$ contains at least one singleton clause, which implies $\sigma(\omega) \leq 1/2$. 


Theorem 1.11 (Spectral Trichotomy Theorem, Constructive Version). For each $k$ and $\delta$:
1. If $\delta$ is a $k$-target for $3\text{CNFS}$, then $\text{kSAT-PROB}_{2,\delta}$ is NP-complete.
2. If $\delta$ is a $k$-target for $2\text{CNFS}$, but not for $3\text{CNFS}$, then $\text{kSAT-PROB}_{3,\delta}$ is NL-complete.
3. In all other cases, $\text{kSAT-PROB}_{2,\delta}$ lies in (DLOGTIME-uniform) $AC^0$.

The hardness results implicit in the claim of the theorem turn out to be easy to prove, mainly due to the fact that the definition of “$\delta$ is a $k$-target for $2\text{CNFS}$ (or $3\text{CNFS}$)” is tailored exactly towards making these results easy-to-prove: Take the above example $\delta = 7/32$ via the above $\omega = \{\{a\}, \{b\}, \{c_1, c_2, c_3\}\}$. To reduce $2\text{SAT}$ to $3\text{SAT-PROB}_{2,\delta}$, on input of $\psi \in 2\text{CNFS}$ (with fresh variables, that is, $\vars(\psi) \cap \vars(\omega) = \emptyset$) output

$$\rho = \{\{a\} \cup d \mid d \in \psi\} \cup \{\{b\} \cup d \mid d \in \psi\} \cup \{\{c_1, c_2, c_3\}\},$$

that is, “add the clauses of $\psi$ to each ‘small’ clause, that is, to each clause in $\omega_{\leq k-1}$.” The important observation is that all satisfying assignments of $\omega$ are also satisfying assignments of $\rho$ (so $\sigma(\rho) \geq \sigma(\omega)$), but there will be additional satisfying assignments of $\rho$ when $\psi$ is satisfiable via some $\alpha$: $\vars(\psi) \rightarrow \{0, 1\}$: This additional satisfying assignment is obtained by merging $\alpha$ with any assignment $\beta$ that witnesses $\{\{c_1, c_2, c_3\}\} \not\subseteq \{\{a\}, \{b\}\}$. The same works for the NP-hardness, now for “$\delta$ is a $k$-target for $2\text{CNFS}$.”

The tricky part are the upper bounds for the last two items. For the last item (the other one is very similar), we wish to show $\text{kSAT-PROB}_{3,\delta} \in AC^0$ when $\delta$ is not a $k$-target for $2\text{CNFS}$.

The initial idea is easy: On input $\phi$, compute the kernel $\kappa$ using Algorithm 1. If $\sigma(\kappa) > \delta$ or $\sigma(\kappa) < \delta$, the same inequality will hold for $\sigma(\phi)$; but in case $\sigma(\kappa) = \delta$, we need to answer the question whether there is a satisfying assignment of $\phi$ that does not also satisfy the kernel. (An NP-machine could easily answer this and this already proves the upper bound of the first statement – but we seek an $AC^0$ upper bound.)

At this point, we use another tool from classical FPT theory: backdoor sets. A (strong) backdoor set for a formula $\omega$ is a set $V$ of variables such that if $\omega \in SAT$, then for all assignments $\beta: V \rightarrow \{0, 1\}$ the formula $\omega|_\beta$ is (highly) tractable, meaning for instance that it is a Horn formula or lies in $2\text{CNFS}$ or lies even in $1\text{CNFS}$.

The central intuition for the $AC^0$ upper bound is that the variables in the kernel $\kappa$ should always form a backdoor set into $1\text{CNFS}$ for $\phi$ when $\delta$ is not a $k$-target for $2\text{CNFS}$: Suppose there is a $\beta: \vars(\kappa) \rightarrow \{0, 1\}$ for which $\phi|_\beta$ contains a clause $d$ with $|d| \geq 2$, meaning that $\vars(\kappa)$ is not a backdoor into $1\text{CNFS}$. Then $\kappa$ must contain a clause $e^*$ of size at most $k$ – 2. Each clause $d \in \phi|_\beta$ results from some clause $e \in \phi \in 3\text{CNFS}$ from which we remove all variables in the kernel, so $|e^*| = |e| - |d| \leq k$ – 2. This shows that $\kappa_{\leq k-2}$ is not empty – meaning that we are in the situation from the above lower bound for $\omega = \kappa$: There is at least one “target clause $e^*$” to which we could add the clauses of $2\text{CNF}$ formulas during a reduction – exactly what we ruled out by assuming that $\delta$ is not a $k$-target for $2\text{CNFS}$. Figure 4 on page 21 visualizes this intuition (for $2\text{CNFS}$ rather than $1\text{CNFS}$, though).

There is a catch, however: To serve as target for a reduction, it does not suffice that a clause of size $k$ – $2$ or less exists in $\omega$. We also need that $\omega_{k-2} \not\models \omega_{\leq k-2}$, that is, that the small clause that we wish to use as target for the reduction is not already implied by the large clauses (and, thus, adding literals to the small clause does not actually give new models and does not raise the satisfaction probability). Fixing this problem is the last step in the algorithm: Instead of just setting $\omega = \kappa$, we let $\omega$ be the small clauses of $\kappa$ plus the links of the large clauses of $\kappa$. Two simple lemmas will show that this $\omega$ has the desired properties: The variables in $\kappa$ form a backdoor into $1\text{CNFS}$ for $\omega$; and $\sigma(\phi) > \delta$ (which is what we wish to decide) if $\delta(\omega) > \delta(\kappa) = \delta$ (which we can decide in $AC^0$ by checking whether $\omega|_\beta \in SAT$ for some $\beta: \vars(\kappa) \rightarrow \{0, 1\}$ with $\beta \not\models \kappa$; and $\vars(\kappa)$ is a strong backdoor into $1\text{CNFS}$).
1.2 Related Work

The history of determining the complexity of the many different variants of the satisfiability problem for propositional formulas dates back all the way to Cook’s original NP-completeness proof [7]. Since then, it has become textbook knowledge that $k$-SAT is in AC$^0$ for $k = 1$, is NL-complete for $k = 2$, and is NP-complete for $k \geq 3$.

Determining whether the number of satisfying assignments of a formula is not just positive, but whether “a lot” of assignments are satisfying, is a quite different problem. Determining whether a majority of assignments are satisfying is a canonical PP-complete problem [10, 13]; and it does not matter whether one considers “strictly more than 1/2” ($\text{SAT-PROB}_{>1/2}$) or “more than or equal to 1/2” or ($\text{SAT-PROB}_{\geq 1/2}$). Indeed, any fixed value different from 1/2 can also be used and it does not matter whether “>” or “≥” is used [13, Theorem 4.1]. Because of the indifference of the complexity to the exact problem definition, it is often a bit vague how the problem “MAJORITY-SAT” is defined, exactly, in a paper (indeed, the common meaning of “majority” in voting suggests that “strictly more than one half” is perhaps the natural interpretation).

Given that the tipping point between “easy” and “hard” satisfaction problems is exactly from $k = 2$ to $k = 3$, it seemed natural to assume that $k$-SAT-PROB$_{>1/2}$ and $k$-SAT-PROB$_{\geq 1/2}$ are also both PP-complete for $k \geq 3$. Indeed, given that computing $\#(\phi)$ for $\phi \in 2CNFS$ is known to be #P-complete [15], even 2SAT-PROB$_{>1/2}$ being PP-complete seemed possible and even natural. It was thus surprising that Akmal and Williams [2] were recently able to show that $k$-SAT-PROB$_{>\delta}$ is in LINTIME holds for all $k$ and $\delta \in \mathbb{Q}$. As pointed out by Akmal and Williams, not only has the opposite generally been believed to hold, this has also been claimed repeatedly (page 1 of [1] lists no less than 15 different papers from the last 20 years that conjecture or even claim hardness of 3SAT-PROB$_{>1/2}$). Just as surprising was the result of Akmal and Williams that while 4SAT-PROB$_{>1/2}$ lies in P, the seemingly almost identical problem 4SAT-PROB$_{>1/2}$ is NP-complete. This has lead Akmal and Williams to insist on a precise notation in [2]: They differentiate clearly between MAJORITY-SAT and GT-MAJORITY-SAT and consider these to be special cases of the threshold problems THR$_{\delta}$-SAT and GT-THR$_{\delta}$-SAT – and all of these problems can arise in a “-kSAT” version. The notations $k$-SAT-PROB$_{>\delta}$ and $k$-SAT-PROB$_{\geq \delta}$ from the present paper are a proposal to further simplify, unify, and clarify the notation, no new problems are introduced.

We will use tools from fpt theory, namely kernels in Algorithm 1 and backdoor sets in Section 4. As computing (especially hitting set) kernels is very well-understood from a complexity-theoretic point of view (see [16] for the algorithmic state of the art and [5] for the upper bounds on the parallel parameterized complexity), we can base proofs on this for $k$-SAT-PROB$_{>\delta} \in \text{AC}^0$ for all $k$ and $\delta$. Of course, different parameterized versions of SAT are studied a lot in fpt theory, see [9] for a starting point, but considering the satisfaction probability as a parameter (as we do in the present paper) is presumably new.

The algorithm of Akmal and Williams in [2] was the main inspiration for the results of the present paper and it shares a number of characteristics with Algorithm 1 (less with Algorithm 3 and none with Algorithm 2, though): Both algorithms search for and then collapse sunflowers. However, without the Spectral Well-Ordering Theorem, one faces the problem that collapsing large sunflowers repeatedly could conceivably lower $\sigma(\phi)$ past $\delta$. To show that this does not happen (without using the Spectral Well-Ordering Theorem) means that one has to redo all the arguments used in the proof of the Spectral Well-Ordering Theorem, but now with explicit parameters and constants and one has to intertwine the algorithmic and the underlying order-theoretic arguments in rather complex ways (just the analysis of the algorithm in [1] takes eleven pages in the main text plus two pages in the appendix).
The majority-of-majority problem for arbitrary CNF formulas, called MAJ-MAJ-SAT in [2], is known to be complete for \( \text{PP}^\text{PF} \) and of importance in “robust” satisfaction probability estimations [6, 12]. The arguments from both the present paper and [2] on \( k\text{-SAT-PROB}_2 \) do not generalize in any obvious way to MAJ-MAJ-\( k \text{-SAT} \): The difficulty lies in the “mixed” clauses that contain both \( X \)- and non-\( X \)-variables. In a clever argument, Akmal and Williams were able to show that for \( k = 2 \) one can “separate” the necessary satisfaction probability estimations for the \( X \)- and non-\( X \)-variables in polynomial time; a feat facilitated by the fact that a mixed size-2 clause must contain exactly one \( X \)-literal and one non-\( X \)-literal. They conjectured that MAJ-MAJ-\( k \text{-SAT} \in \text{P} \) holds for all \( k \) (which is indeed the case by Corollary 3.10), but point out that it is unclear how (or whether) their algorithm can be extended to larger \( k \). The approach taken in the present paper (via locality arguments) seems quite different and not directly comparable.

## 2 Order-Theoretic Results

There is a sharp contrast between the structural properties of the “full” spectrum of satisfaction probabilities of arbitrary propositional formulas and the spectrum of values \( k \text{CNF} \) formulas can have. The full spectrum \( \text{cnfs-}\sigma\text{-SPECTRUM} = \{ \sigma(\phi) \mid \phi \in \text{CNF} \} \) is, well, “full” as it is the set of all dyadic rationals between 0 and 1 (recall \( \mathbb{D} = \{ m/2^c \mid m, c \in \mathbb{Z} \} \)).

\[ \triangleright \text{Lemma 2.1.} \text{ cnfs-}\sigma\text{-SPECTRUM} = \mathbb{D} \cap [0, 1]. \]

**Proof.** For any propositional formula \( \phi \) we have, by definition, \( \sigma(\phi) = m/2^c \) for \( m = \#(\phi) \) and \( e = |\text{vars}(\phi)| \). For the other direction, let \( e \in \mathbb{N} \) and \( m \in \{0, \ldots, 2^e\} \). Consider the truth table over the variables \( X = \{ x_1, \ldots, x_e \} \) in which the first \( m \) lines are set to 1 (“models”) and the rest are set to 0 (“not models”). Then each CNF formula \( \phi \) with \( \text{vars}(\phi) = X \) having this truth table has exactly \( m \) models and, hence, \( \sigma(\phi) = m/2^c \). \( \square \)

By the lemma, cnfs-\( \sigma \)-SPECTRUM is a dense subset of the real interval \([0, 1]\), it is not closed topologically, and it is order-isomorphic to \( \mathbb{Q} \cup \{-\infty, \infty\} \) with respect to both \( < \) and \( > \). We will soon see that the properties of each \( k \text{CNF-}\sigma\text{-SPECTRUM} \) could hardly be more different: They are nowhere-dense, they are closed, and they are well-ordered. Of these properties, the well-orderedness is the most important one both for algorithms in later sections and because the other properties follow from the well-orderedness rather easily.

To get a better intuition about the spectra, let us have a closer look at the first two, 1cnfs-\( \sigma \)-SPECTRUM and 2cnfs-\( \sigma \)-SPECTRUM. The first is simple:

\[ 1\text{cnfs-}\sigma\text{-SPECTRUM} = \{ 1, \frac{1}{2}, \frac{4}{7}, \frac{1}{8}, \frac{1}{16}, \ldots \} \cup \{ 0 \} \] (1)

as a 1CNF formula (a conjunction of literals) has a satisfaction probability of the form \( 2^{-e} \) or is 0. Readers familiar with order theory will notice immediately that 1cnfs-\( \sigma \)-SPECTRUM is order-isomorphic to the ordinal \( \omega + 1 \) with respect to \( > \). The spectrum 2cnfs-\( \sigma \)-SPECTRUM is already much more complex:

\[ 2\text{cnfs-}\sigma\text{-SPECTRUM} = \{ 1, \frac{3}{4}, \frac{5}{8}, \frac{9}{16}, \frac{17}{32}, \frac{33}{64}, \ldots \} \cup \{ \frac{1}{2}, \frac{15}{32} \} \cup \{ \ldots \} \] (2)

where \( \{ \ldots \} \) contains only numbers less than 15/32. To see that this is, indeed, the case, observe that the formulas \( \{ \{ a, x_1 \}, \{ a, x_2 \} \} \), \( \{ \{ a, x_1 \}, \{ a, x_2 \}, \{ a, x_3 \} \} \), \ldots show that every number of the form \( 1/2 + 2^{-e} \) is in the spectrum. Furthermore, there are no other numbers larger than 1/2 in the spectrum as the first formula with two variable-disjoint clauses has a satisfaction probability of \( \sigma(\{ \{ a, b \}, \{ c, d \} \}) = 9/16 \) which we happen
to have already had; and adding any additional clause makes the probability drop to at most \( \sigma(\{(a, b), \{c, d\}, \{c, e\}\}) = 3/4 \cdot 5/8 = 15/32 \). Below this, the exact structure of 2CNFS-\( \sigma \)-SPECTRUM becomes ever more complex as we get nearer to 0 and it is unclear what the order-type of 2CNFS-\( \sigma \)-SPECTRUM with respect to \( \sigma \) actually is (an educated guess is \( \omega^\omega + 1 \)).

Our aim in the rest of this section is to prove the Spectral Well-Ordering Theorem, Theorem 1.1, by which all \( k \)CNFS-\( \sigma \)-SPECTRUM are well-ordered by \( \sigma \). The surprisingly short proof, presented in Section 2.2, will combine results from the following Section 2.1 on some simple properties of well-orderings with some simple properties of \( \sigma(\phi) \) for \( k \)CNF formulas \( \phi \). The theorem implies the existence of spectral gaps below each \( \delta \) in the spectra, but the proof does not provide us with any quantitative information about the sizes of these gaps. It is possible, but challenging to obtain bounds on the sizes of spectral gaps; please see the technical report version [14] of this paper for the (very) technical details.

### 2.1 Well-Orderings and Their Properties

Well-orderings are a basic tool of set theory, but for our purposes only a very specific type of orderings will be of interest (namely only sets of non-negative reals with the strictly-greater-than relation as the only ordering relation). For this reason, we reserve the term “well-ordering” only for the following kind of orderings:

**Definition 2.2.** A set \( X \subseteq \mathbb{R}^{\geq 0} \) is well-ordered (by \( \sigma \)) if there is no sequence \( (x_i)_{i \in \mathbb{N}} \) with \( x_i \in X \) for all \( i \) and \( x_0 < x_1 < \cdots \). Let \( \text{WO} \) denote the set of all (such) well-ordered sets.

There is extensive literature on the properties of well-orderings in the context of classical set theory, see for instance [11] as a starting point. We will need only those stated in the following lemma, where the first items are standard, while the last are specific to the present paper. For \( X, Y \subseteq \mathbb{R}^{\geq 0} \) let \( X + Y \) denote \( \{x + y \mid x \in X, y \in Y\} \) and \( n \cdot X = X + \cdots + X \) (of course, \( n \) times). Note that “sequence” always means “infinite sequence” in the following and that “strictly increasing” means \( x_i < x_j \) for \( i < j \) while just “decreasing” (without the “strictly”) means \( x_i \geq x_j \) for \( i < j \).

**Lemma 2.3.**

1. Let \( X \in \text{WO} \). Then \( X \) contains a largest element.
2. Let \( Y \subseteq X \in \text{WO} \). Then \( Y \in \text{WO} \).
3. Let \( X, Y \in \text{WO} \). Then \( X \cup Y \in \text{WO} \). Thus, \( \text{WO} \) is closed under finite unions.
4. Let \( (X_p)_{p \in \mathbb{N}} \) with \( X_p \in \text{WO} \) for all \( p \) and \( \lim_{p \to \infty} \max X_p = 0 \). Then \( \bigcup_{p \in \mathbb{N}} X_p \in \text{WO} \).
5. Let \( X \in \text{WO} \) and let \( (x_i)_{i \in \mathbb{N}} \) be an arbitrary sequence of \( x_i \in X \). Then there is an infinite \( I \subseteq \mathbb{N} \) such that \( (x_i)_{i \in I} \) is decreasing (that is, \( x_i \geq x_j \) for \( i < j \) and \( i, j \in I \)).
6. Let \( X, Y \in \text{WO} \). Then \( X + Y \in \text{WO} \). Thus, \( \text{WO} \) is closed under finite sums.
7. Let \( X \in \text{WO} \) and \( n \in \mathbb{N} \). Then \( n \cdot X \in \text{WO} \).

**Proof.**

1. There would otherwise be an (infinite) strictly increasing sequence in \( X \).
2. Any strictly increasing sequence in \( Y \) would be a strictly increasing sequence in \( X \).
3. Any strictly increasing sequence in \( X \cup Y \) would contain a strictly subsequence in \( X \) or \( Y \).
4. Assume there is a sequence \( 0 < x_0 < x_1 < x_2 < \cdots \) where \( x_i \in \bigcup_{p \in \mathbb{N}} X_p \) holds for all \( i \).
   As the maxima of the \( X_p \) tend towards 0, there is some \( q \) such that \( x_0 > \max X_p \) holds for all \( p \geq q \). In particular, the whole sequence \( (x_i)_{i \in \mathbb{N}} \) contains only elements of \( \bigcup_{p \leq q} X_p \).
   By the previous item, this is well-ordered, contradicting that it contains an infinite strictly increasing sequence.
5. The set \( \{x_i \mid i > 0\} \) is a subset of \( X \) and must hence contain a maximal element \( x_{i_0} \) by the first item. Then \( \{x_i \mid i > i_0\} \subseteq X \) must contain a maximal element \( x_{i_1} \) for some \( i_1 > i_0 \). Next, consider \( \{x_i \mid i > i_1\} \subseteq X \) and let \( x_{i_2} \) for some \( i_2 > i_1 \) be a maximal element. In this way, for \( I = \{i_0, i_1, \ldots\} \) we get an infinite subsequence \( (x_i)_{i \in I} \) that is clearly (not necessarily strictly) decreasing as each chosen element was the maximum of all following elements.

6. Suppose there is a sequence \( z_0 < z_1 < z_2 < \ldots \) of numbers \( z_i \in X + Y \). Then for each \( i \) there must exist \( x_i \in X \) and \( y_i \in Y \) with \( z_i = x_i + y_i \). By the previous item there is a decreasing subsequence \( (x_i)_{i \in I} \) for some infinite \( I \). Then \( (y_i)_{i \in I} \) is an infinite strictly increasing sequence in \( Y \) as for any \( i, j \in I \) with \( i < j \) we have \( y_i = z_i - x_i \leq z_j - x_j < z_j - x_j = y_j \). This contradicts \( Y \in \text{WO} \).

7. This follows immediately from the previous item.

### 2.2 Proof of the Spectral Well-Ordering Theorem

For the proof of the Spectral Well-Ordering Theorem, besides the Packing Probability Lemma (Lemma 1.8 from the introduction) we will need the notion of expansions, which allow us to express the satisfaction probability of a formula as a sum of satisfaction probabilities of restricted formulas. The definition is based on the well-known unit rule: For a formula \( \phi \in \text{cnfs} \), let \( \text{unitrule}(\phi) \) be obtained by removing, for each clause \( \{l\} \in \phi \) containing a single literal \( l \), all other clauses from \( \phi \) containing \( l \) (so, perhaps a bit non-standard, we leave the singleton clause \( \{l\} \), which “triggered” the unit rule, in the formula) and removing all occurrences of the negated literal from all remaining clauses.

**Definition 2.4.** For a set \( V \) of variables, \( \beta : V \to \{0, 1\} \) and \( \phi \in \text{cnfs} \), the restriction \( \phi|_{\beta} \) of \( \phi \) to \( \beta \) is \( \text{unitrule}(\phi \cup \{\{v\} \mid v \in V, \beta(v) = 1\} \cup \{\{\neg v\} \mid v \in V, \beta(v) = 0\}) \).

As an example, for \( \phi = \{\{a, b\}, \{\neg b, c, \neg f\}, \{d, e, f, g\}\} \) and \( \beta : \{b, c, h\} \to \{0, 1\} \) with \( \beta(b) = 1 \) and \( \beta(c) = \beta(h) = 0 \) we have \( \phi|_{\beta} = \{\{a, b\}, \{\neg b, c, \neg f\}, \{d, e, f, g\}\} \cup \{\{b\}, \{\neg c\}, \{\neg h\}\} = \{\{\neg f\}, \{d, e, f, g\}, \{b\}, \{\neg c\}, \{\neg h\}\} \).

Note that we apply the unit rule only once to the initial singleton clauses, we do not do unit propagation, which is \( \text{P} \)-hard and thus computationally too expensive in later contexts. The importance of restrictions for our purposes lies in two simple lemmas:

**Lemma 2.5.** Let \( \phi \in \text{cnfs} \) and \( V \) be some variables. Then the set of models of \( \phi \) over the variables \( V \cup \text{vars}(\phi) \) is exactly the disjoint union of the models of \( \phi|_{\beta} \) for \( \beta : V \to \{0, 1\} \).

**Proof.** Each satisfying assignment \( \alpha : V \cup \text{vars}(\phi) \to \{0, 1\} \) of \( \phi \) satisfies \( \phi|_{\beta} \) for the assignment \( \beta \) that agrees with \( \alpha \) on the variables in \( V \), but \( \alpha \) satisfies no \( \phi|_{\beta'} \) for \( \beta' \neq \beta \).

**Corollary 2.6.** Let \( \phi \in \text{cnfs} \) and \( V \) be some variables. Then \( \sigma(\phi) = \sum_{\beta : V \to \{0, 1\}} \sigma(\phi|_{\beta}) \).

**Lemma 2.7.** Let \( \phi \in \text{kcns} \) for \( k \geq 2 \) and let \( \pi \subseteq \phi \) be a maximal packing. Then \( \phi|_{\beta} \in (k-1)\text{cnfs} \) for all \( \beta : \text{vars}(\pi) \to \{0, 1\} \).

**Proof.** In \( \phi|_{\beta} \), we either remove a clause or remove at least one literal from it as \( \text{vars}(\pi) \) intersects \( \text{vars}(c) \) for all \( c \in \phi \) (as \( \pi \) would not be maximal, otherwise).

### Proof of the Spectral Well-Ordering Theorem, Theorem 1.1

By induction on \( k \). The base case is \( k = 1 \) where \( 1\text{cnfs}-\sigma\text{-spectrum} = \{1, \frac{1}{2}, \frac{1}{3}, \ldots\} \cup \{0\} = \{2^{-i} \mid i \in \mathbb{N} \cup \{\infty\}\} \), which is clearly well-ordered (with order type \( \omega + 1 \)). For the inductive step from \( k-1 \) to \( k \), we show that

\[
\text{kcnfs-}\sigma\text{-spectrum} \subseteq \bigcup_{p \in \mathbb{N}} (2^{kp} \cdot (k-1)\text{cnfs-}\sigma\text{-spectrum}) \cap [0, (1 - 2^{-k})p].
\]
This will prove $\text{kCNFS-} \sigma \text{-spectrum} \in \text{WO}$ as $(k - 1)\text{CNFS-} \sigma \text{-spectrum} \in \text{WO}$ by the induction hypothesis, and thus $2^{kp} \cdot (k - 1)\text{CNFS-} \sigma \text{-spectrum} \in \text{WO}$ as a finite sum of well-orderings. By intersecting this with the ever-smaller intervals $[0, (1 - 2^{-k})^p]$, we still get elements of $\text{WO}$ by item 2 of Lemma 2.3 and can then apply item 4 to get that the infinite union lies in $\text{WO}$.

It remains to prove the inclusion. Let $x \in \text{kCNFS-} \sigma \text{-spectrum}$ be witnessed by $\phi \in \text{CNFS}$, that is, $x = \sigma(\phi)$. Let $\pi$ be a maximal packing $\pi \subseteq \phi$ and let $p = |\pi|$. By the Packing Probability Lemma, $\sigma(\phi) \leq (1 - 2^{-k})^p$. By Corollary 2.6, we have $\sigma(\phi) = \sum_{\beta \in \text{vars}(\pi) \to \{0, 1\}} \sigma(\phi|\beta)$ and by Lemma 2.7, each $\phi|\beta$ is a $(k - 1)\text{CNF}$ formula. Thus, $\sigma(\phi)$ is the sum of at most $2^{kp}$ values from $(k - 1)\text{CNFS-} \sigma \text{-spectrum}$, proving that we have $x \in [2^{kp}, (k - 1)\text{CNFS-} \sigma \text{-spectrum}] \cap [0, (1 - 2^{-k})^p]$. △

3 Algorithmic Results

Having established that the spectra $\text{kCNFS-} \sigma \text{-spectrum}$ are well-ordered by $>$ in the previous section, we now turn our attention to the algorithmic aspects of deciding whether $\sigma(\phi) \geq \delta$ holds (the question of whether $\sigma(\phi) > \delta$ holds will be discussed in the next section). The focus will be less on the exact complexity of these algorithm (they can all be implemented by $AC^0$ circuits, sometimes trivially, sometimes with a bit of effort), but more on how the algorithms work. We will touch on “practical” considerations only very briefly in the following, as they do not lie at the heart of this paper; some ideas towards optimizations, implementations, and practical heuristics can be found in the technical report version [14].

This section includes two “excursions” beyond the three main algorithms. First, an interesting corollary of the algorithmic analysis of the first algorithm will be another structural result concerning the spectra $\text{kCNFS-} \sigma \text{-spectrum}$, though not an order-theoretic one, but a topological one: The spectra are closed, meaning that for every converging sequence $\sigma(\phi_1),\sigma(\phi_2),\ldots$ with $\phi_i \in \text{CNF}$ there is some $\phi \in \text{CNF}$ with $\lim_{i\rightarrow\infty} \sigma(\phi_i) = \sigma(\phi)$. Another way of saying this is that for every number $\delta \notin \text{kCNFS-} \sigma \text{-spectrum}$ there is an $\epsilon > 0$ with $(\delta - \epsilon, \delta + \epsilon) \cap \text{kCNFS-} \sigma \text{-spectrum} = \emptyset$. Second, we will show that the Threshold Locality Lemma, which underlies the second algorithm, can also be used to reduce certain satisfaction probability threshold problems to model checking problems. This will allow us to “chip away” one “majority-of-” in any majority-of-majority-of-...-majority problem. In particular, we will get a proof of the Akmal–Williams conjecture $\text{maj-maj-}k\text{SAT} \in \text{P}$.

3.1 The Sunflower-Collapsing Algorithm

Our first new algorithm for deciding whether $\sigma(\phi) \geq \delta$ holds for a given $\phi \in \text{CNF}$ and fixed $\delta$ is based on computing “sunflower kernels.” The name “sunflower” comes from classical combinatorics, the name “kernel” comes from fixed-parameter tractability (FPT) theory. Note that we just use kernels as a “tool” without making formal statements in the sense of FPT theory, but see the technical report version [14] for some first formal statements.

Recall from Definition 1.9 that a sunflower with core $c$ is a formula $\psi \in \text{CNF}$ such that $c \subseteq e$ holds for all clauses $e \in \psi$ and such that any two different $e, e' \in \psi$ we have $\text{vars}(e) \cap \text{vars}(e') = \text{vars}(c)$, that is, the clauses “agree on the literals in $c$, but are variable-disjoint otherwise”. An example of a size-3 sunflower with core $\{a, \neg b, c\}$ is $\{\{a, \neg b, c\}, \{a, \neg b, \neg d\}, \{a, \neg b, f\}\}$.

Sunflowers play a key role in FPT theory, especially in the computation of kernels for the hitting set problem [9] and related problems: Suppose that for a given formula $\phi$ we want to find a size-$k$ set $V$ of variables such that for each clause $e \in \phi$ we have $\text{vars}(e) \cap V \neq \emptyset$
(each clause is “hit” by $V$). Then if there is a sunflower $\psi \subseteq \phi$ of size $h + 1$ in $\phi$ with some core $c$, any size-$h$ hitting set $V$ must hit $c$ since, otherwise, we would need $h + 1$ variables to hit the “petals” of the sunflower (we would need to have $V \cap (\text{vars}(c) \setminus \text{vars}(e)) \neq \emptyset$ for $h + 1$ pairwise disjoint sets $\text{vars}(e) \setminus \text{vars}(c)$). This means that $\phi$ has a size-$h$ hitting set iff $(\phi \setminus \psi) \cup \{c\}$ has (indeed, iff $(\phi \setminus \text{link}_\phi(c)) \cup \{c\}$ has, where $\text{link}_\phi(c) = \{e \in \phi \mid e \subseteq c\}$). Most importantly, applying this reduction rule “as often as possible” leads to a formula whose size is bounded by a constant depending only on $h$, not on the original formula (this is known as a “kernelization” in fixed-parameter theory). The reason for this size bound is the following Sunflower Lemma (rephrased in terms of positive formulas rather than hypergraphs, as would be standard, where a positive formula is a formula without negations):

> **Fact 3.1** (Sunflower Lemma, [8]). Every positive $\phi \in \text{kcnfs}$ with more than $h^k \cdot k!$ clauses contains a sunflower of size $h + 1$.

The “positive” in the statement is due to the fact that in combinatorics sunflowers usually do not care about the “sign” of the variables (whether or not it is negated). In particular, for a formula $\phi$ let $\phi'$ be the formula where all negations are simply removed. Then the Sunflower Lemma tells us that if $\phi'$ is sufficiently large, then it has a large sunflower $\psi \subseteq \phi'$ with core $c$. This large sunflower does not necessarily become a large sunflower of the original $\phi$ if we just reinsert the negations. While this makes no difference for the petals outside the core, there may now suddenly be up to $2^{|c|}$ different versions of the core. However, for the version of this core that is present in the maximum number of petals, the number of these petals is at least a fraction of $1/2^{|\psi|} \geq 1/2^k$ of the size of the “unsigned” sunflower. This yields the following corollary:

> **Corollary 3.2.** Every $\phi \in \text{kcnfs}$ with more than $(2h)^k \cdot k!$ clauses contains a sunflower of size $h + 1$.

The just-described kernel algorithm for the hitting set problem (“as long as there is a sufficiently large sunflower, remove it and add the core instead”) also turns out to allow us to decide whether $\sigma(\phi) \geq \delta$ holds. The detailed algorithm is Algorithm 1 from page 6. Its important properties are summarized in the following lemma:

> **Lemma 3.3.** For each fixed $k$ and $\delta$, in Algorithm 1 from page 6

1. the while loop will terminate after a number of iterations that is linear in the size of $\phi$,
2. the size $|\kappa|$ will be bounded by a fixed number that depends only on $k$ and $\delta$, and
3. $\sigma(\phi) \geq \delta$ iff $\sigma(\kappa) \geq \delta$ will hold (the “kernel property”) and, thus, the output of the algorithm $\text{SUNFLOWER-COLLAPSING}(\phi)$ will be correct.

**Proof.** For item 1, each iteration of the while loop reduces the size of $\phi$, so after a linear number of iterations the loop must stop. (Note that finding large sunflowers is a bit of an art and there is extensive literature on how to do this efficiently, see [5, 9, 16] for starting points. But as the size $h$ is fixed in our case, we could even brute-force the search here.)

For item 2, we use Corollary 3.2, which states that as long as there are more than $(2h)^k \cdot k!$ clauses in $\phi$, there is still a sunflower and, hence, the while loop will not have ended. Thus, $|\kappa| \leq (2h)^k k!$ and this is clearly a constant as $h$ is a constant depending only on $\delta$.

For item 3, we need to show that for $\phi' = (\phi \setminus \text{link}_\phi(c)) \cup \{c\}$, where $c$ is the core of a size-$(h + 1)$ sunflower $\psi$ in $\phi$, we have $\sigma(\phi) \geq \delta$ iff $\sigma(\phi') \geq \delta$. As already pointed out in the introduction, the probability $\sigma(\phi) - \sigma(\phi')$ that an assignment $\beta$ satisfies $\phi$, but not $\phi'$, is at most $\sigma(\{e \in c \mid e \in \text{link}_\phi(c)\})$. As the link contains a size-$(h + 1)$ sunflower,
\{e \wedge c \mid e \in \text{link}_0(c)\} contains a size-(h + 1) packing and by the Packing Probability Lemma we get \(\sigma(\phi) - \sigma(\phi') \leq (1 - 2^{-k})^{h+1} < (1 - 2^{-k})^h = \text{spectral-gap}_{\text{cnfs}}(\delta)\). By the No Tunneling Lemma, Lemma 1.10, we get the claim.

Excursion 1: The Spectra Are Topologically Closed. While the sunflower collapsing algorithm provides a surprisingly simple method of deciding whether \(\sigma(\phi) \geq \delta\) holds, it can also be seen as a purely combinatorial statement:

Lemma 3.4. For every \(\delta \in [0, 1]\) and \(k\) there is a size \(s\) such that for every formula \(\phi \in \text{kcnsfs}\) with \(\sigma(\phi) \geq \delta\) there is a formula \(\kappa \in \text{kcnsfs}\) of size \(|\kappa| \leq s\) with \(\sigma(\phi) \geq \sigma(\kappa) \geq \delta\).

\[\text{Proof.}\] For fixed \(\delta\) and \(k\), let \(s\) be the maximum size of the output \(\kappa\) of KERNELIZE(\(\phi\)) in Algorithm 1. The kernel always has the property \(\sigma(\phi) \geq \sigma(\kappa)\). As shown in the proof of the second item of Lemma 3.3, \(s \leq (2\log_{1 - 2^{-k}}(\text{spectral-gap}_{\text{cnfs}}(\delta)))^k k!\). Finally, by the third item, \(\sigma(\phi) \geq \delta\) implies \(\sigma(\kappa) \geq \delta\).

This lemma provides us with an easy way of showing that \(\text{kcnsfs-}\sigma\text{-spectrum}\) is topologically closed, which means that its complement is an open set. Note that this does not follow from the fact that the spectra are well-ordered as the set \(\{1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots\}\) is well-ordered, but not closed (it misses 0).

Lemma 3.5. Let \(\Phi \subseteq \text{kcnsfs}\). Then \(\inf\{\sigma(\phi) \mid \phi \in \Phi\} \in \text{kcnsfs-}\sigma\text{-spectrum}\).

\[\text{Proof.}\] Let \(\delta = \inf\{\sigma(\phi) \mid \phi \in \Phi\}\). Then there must be a sequence \((\phi_0, \phi_1, \phi_2, \ldots)\) with \(\phi_i \in \Phi\) with \(\lim_{i \to \infty} \sigma(\phi_i) = \delta\). Consider the sequence \((\kappa_0, \kappa_1, \kappa_2, \ldots)\) where each \(\kappa_i\) is the formula from Lemma 3.4 for \(\phi_i\). Then, clearly, \(\lim_{i \to \infty} \sigma(\kappa_i) = \delta\). If necessary, rename the variables in each \(\kappa_i\), to that they are \(\{v_1, \ldots, v_q\}\) for \(q = k \cdot s\), where \(s\) is the constant form Lemma 3.4, and note that this is always possible. Then \(K := \{\kappa_i \mid i \in \mathbb{N}\}\) is a finite set as there are only finitely many different cnf formulas over the variables \(\{v_1, \ldots, v_q\}\). This means that there is some \(\kappa^* \in K\) with \(\sigma(\kappa^*) = \min\{\sigma(\kappa) \mid \kappa \in K\}\). Then \(\sigma(\kappa^*) = \delta\) must hold and \(\kappa^* \in \text{kcnsfs}\) witnesses \(\delta \in \text{kcnsfs-}\sigma\text{-spectrum}\).

Corollary 3.6. \(\text{kcnsfs-}\sigma\text{-spectrum}\) is topologically closed for each \(k\).

3.2 The Threshold Locality Lemma and Algorithm

A second algorithm for showing \(ksat\text{-PROB}_{\geq \delta} \in \text{AC}^0\) is based on Lemma 1.5 from the introduction, which states: For each \(k\) and each \(\delta \in [0, 1]\) there is a \(C \in \mathbb{N}\) so that for all \(\phi \in \text{kcnsfs}\) we have \(\sigma(\phi) \geq \delta\), iff \(\sigma(\psi) \geq \delta\) holds for all \(\psi \subseteq \phi\) with \(|\psi| \leq C\). The proof, which was already sketched in the introduction, is surprisingly easy:

Proof of the Threshold Locality Lemma (Lemma 1.5). First note that \(\psi \subseteq \phi\) clearly implies \(\sigma(\psi) \geq \sigma(\phi)\) (as \(\phi\) has “just more clauses” it is “harder to satisfy”). Thus, \(\sigma(\phi) \geq \delta\) immediately implies \(\sigma(\psi) \geq \delta\) for all \(\psi \subseteq \phi\), regardless of their size.

For the other direction, suppose \(\sigma(\phi) < \delta\) holds. We must show that there is some \(\psi \subseteq \phi\) with \(\sigma(\psi) < \delta\) and \(|\psi| \leq C\) for a constant \(C\). We claim that we can always find such a \(\psi\) when we set

\[
C := (2(1 + \log_{1 - 2^{-k}}(\text{spectral-gap}_{\text{cnfs}}(\delta))))^k \cdot k!.
\]
To find such a $\psi$, starting with $\psi_1 = \phi$, as long as possible, pick a clause $e \in \psi_i$ such that $\sigma(\psi_i \setminus \{e\}) < \delta$ still holds and set $\psi_{i+1} = \psi_i \setminus \{e\}$. As we have $\sigma(\psi_1) < \delta$ and as $\sigma(\emptyset) = 1 \geq \delta$, the process must end after $s \leq |\phi|$ steps with some formula $\psi := \psi_s$. Then $\sigma(\psi) < \delta$, $\psi \subseteq \phi$, and for all clauses $e \in \psi$ we have $\sigma(\psi \setminus \{e\}) \geq \delta$.

Suppose we had $|\psi| > C$. By Corollary 3.2 to the Sunflower Lemma, $\psi$ must then contain a sunflower $\rho \subseteq \psi$ of size $|\rho| \geq g + 2$. Let $c$ be the core of $\rho$ and let $a \in \rho$ be arbitrary. Then $\psi \setminus \{a\}$ still contains a sunflower (namely $\rho \setminus \{a\}$) of size at least $g + 1$. Now consider an assignment $\beta$ with $\beta \models \psi \setminus \{a\}$, but $\beta \not\models \psi$. Then $\beta \not\models c$ as, otherwise, $\beta \models \psi$ would hold. Thus, $\beta$ must be a model of the packing $\{e \setminus c \mid e \in \rho \setminus \{a\}\}$ and hence, by the Packing Probability Lemma, $\Pr_{\beta}[\beta \models \psi \setminus \{a\}, \beta \not\models \psi] \leq (1 - 2^{-k})^{\rho \setminus \{a\}} \leq (1 - 2^{-k})^{g+1} < (1 - 2^{-k}) C^{\rho} = \text{spectral-gap}_{\text{cnfs}}(\delta)$. This implies that $\sigma(\psi \setminus \{a\}) - \sigma(\psi)$ is smaller than the spectral gap of $\delta$ and by the No Tunneling Lemma we get $\sigma(\psi \setminus \{a\}) < \delta$; contradicting that we could no longer remove clauses from $\psi$ without raising the satisfaction probability to at least $\delta$.

As pointed out in the introduction, Lemma 1.5 clearly implies that Algorithm 2 is correct; we just state this once more for reference:

\begin{quote}
\textbf{Lemma 3.7.} For each fixed $k$ and $\delta$, on every input $\phi \in \text{cnfs}$ the output of Algorithm 2 from page 7 is correct.
\end{quote}

Excursion 2: Solving the Majority-of-Majority Problem. Besides the simple just-sketched algorithm for solving $\text{ksat-prob}_{2,3}$ efficiently, the Threshold Locality Lemma has another surprising and highly nontrivial consequence: The lemmas lies at the heart of an algorithm for solving the majority-of-majority problem efficiently for $\text{cnfs}$. Recall that for this problem we are given a formula $\phi \in \text{cnfs}$ with $\text{vars}(\phi) \subseteq X \cup Y$ for two disjoint, infinite sets (“sorts”) $X$ and $Y$ of variables. The question is whether for at least half of all possible assignments $\beta : X \rightarrow \{0, 1\}$ at least half of all possible extensions $\beta' : X \cup Y \rightarrow \{0, 1\}$ (meaning $\beta'(x) = \beta(x)$ for all $x \in X$) make $\phi$ true, that is, $\beta' \models \phi$.

While it is tempting to try to solve this problem by focusing on the statement “at least half of all possible assignments $\beta : X \rightarrow \{0, 1\}$” initially, it turns out that it is the statement “at least half of all possible extensions $\beta' : X \cup Y \rightarrow \{0, 1\}$” that we must address first. Our objective will be to replace any formula $\phi \in \text{cnfs}$ with $\text{vars}(\phi) \subseteq X \cup Y$ by a formula $\omega \in \text{cnfs}$ with $\text{vars}(\omega) \subseteq X$ such that $\omega$ “encodes the validity of the second statement for every $\beta$.” The following definition and lemma make these ideas precise (and generalize them to values of $\delta$ other than 1/2):

\begin{quote}
\textbf{Definition 3.8.} Let $\phi \in \text{cnfs}$ with $\text{vars}(\phi) \subseteq X \cup Y$ for disjoint sets $X$ and $Y$. Let $\beta : X \rightarrow \{0, 1\}$. We write $\phi/\beta$ for the formula resulting from $\phi$ when we remove all clauses that contain an $X$-literal $l$ (so $l = x$ or $l = \neg x$ for some $x \in X$) with $\beta(l) = 1$ and where we remove all remaining $X$-literals from all remaining clauses.
\end{quote}

Note that $\text{vars}(\phi/\beta) \subseteq Y$ and that we “almost” have $\phi|_\beta = \phi/\beta$, but the difference is that in $\phi|_\beta$ we have an additional singleton clause for each $x \in X$, whereas $\phi/\beta$ contains none. As an example, for $\phi = \{(x_1, \neg x_2, y_1), (\neg x_1, y_2, \neg y_3), (\neg x_2), (y_1)\}$ and $\beta : \{x_1, x_2\} \rightarrow \{0, 1\}$ with $\beta(x_1) = \beta(x_2) = 1$, we have $\phi/\beta = \{[x_1, \neg x_2, y_1], [\neg x_1, y_2, \neg y_3], [\neg x_2], [y_1]\} = \{[\neg y_3], \emptyset, [y_1], [x_1], [x_2]\}$. The notation makes handling the satisfaction probabilities of extensions easier due to the following simple connection:

$$\sigma(\phi/\beta) = \Pr_{\beta'}[X \cup Y \rightarrow \{0, 1\}, \beta' \models \phi].$$

Note that $\text{maj-maj-ksat} = \{\phi \in \text{cnfs} \mid \Pr_{\beta, X \rightarrow \{0, 1\}}[\sigma(\phi/\beta) \geq 1/2] \geq 1/2\}$.
Lemma 3.9 (Threshold Encoding Lemma). For each \( k \) and \( \delta \in [0, 1] \) there are a number \( l \) and an AC^0 circuit family that maps every formula \( \phi \in k\text{CNF} \) with \( \text{vars}(\phi) \subseteq X \cup Y \) to a formula \( \omega \in \text{lCNFS} \) with \( \text{vars}(\omega) \subseteq X \) such that for all \( \beta \): \( X \rightarrow \{0, 1\} \) we have

\[
\sigma(\phi/\beta) \geq \delta \iff \beta \models \omega.
\]

Proof. Consider the formula \( \rho \) that results from \( \phi \) if we simply remove all \( X \)-variables from all clauses (so \( \text{vars}(\rho) \subseteq Y \)). This formula is, in a sense, the “worst case” of what \( \phi/\beta \) could look like regarding the satisfaction probability: \( \rho \) is the formula where no clause is already satisfied by the assignment \( \beta \), leaving a maximal number of clauses that need to be satisfied. Note that \( \phi/\beta \subseteq \rho \) and, thus, if \( \sigma(\rho) \geq \delta \) happens to hold, we have \( \sigma(\phi/\beta) \geq \delta \) for all \( \beta \) and could set \( \omega \) to an arbitrary tautology. The interesting question is, thus, what happens when \( \sigma(\rho) < \delta \): Which \( \beta \) will remove enough clauses from \( \rho \) to raise the probability above \( \delta \)?

To answer this question (and to turn it into a formula \( \omega \)), we use the Threshold Locality Lemma. By this lemma, there is a constant \( C \) such that we have \( \sigma(\phi/\beta) \geq \delta \) iff for every \( \psi \subseteq \phi/\beta \) with \( |\psi| \leq C \) we have \( \sigma(\psi) \geq \delta \). In particular, for a given \( \psi \subseteq \rho \) with \( \sigma(\psi) < \delta \) we must have \( \psi \not\subseteq \phi/\beta \) to have a chance that \( \sigma(\phi/\beta) \geq \delta \) holds. Now, \( \psi \not\subseteq \phi/\beta \) means that there must be at least one clause \( c \in \psi \) that is not contained in \( \phi/\beta \). By Definition 3.8 this is the case iff there is a clause \( c \in \psi \) such that for all \( d \in \phi \) from which \( c \) resulted (recall that each clause in \( \rho \) is obtained from some clause of \( \phi \) by removing all \( X \)-variables) at least one \( X \)-literal in \( d \) is set to 1 by \( \beta \) (because, then, \( c \) is not added to \( \phi/\beta \)).

To summarize, we have two conditions to check:

1. For every \( \psi \subseteq \rho \) with \( |\psi| \leq C \) and \( \sigma(\psi) < \delta \) we must have \( \psi \not\subseteq \phi/\beta \), which is the case iff
2. there is a clause \( c \in \psi \) such that for all \( d \in \phi \) from which \( c \) resulted, at least one \( X \)-literal in \( d \) is set to 1 by \( \beta \).

It turns out, we can express these conditions using a single formula \( \omega \in \text{lCNFS} \). Let us start with the second condition for a fixed \( \psi \subseteq \rho \) and let us try to find a single formula \( \omega_{\psi} \in \text{lCNFS} \) for some \( l \) expressing it. The condition is clearly a disjunction (“there is a clause”) over all \( c \in \psi \) of the following \( k\text{CNF} \) formulas \( \omega_{\psi,c} \) (“such that for all” is expressed by the union):

\[
\omega_{\psi,c} = \bigcup_{d \in \phi, c \subseteq d, \text{vars}(d) \setminus X = \text{vars}(c)} \{d \setminus c\}.
\]

We can turn the disjunction of the at most \( C \) many \( \omega_{\psi,c} \in \text{kCNFS} \) into a single conjunction \( \omega_{\psi} \in \text{lCNF} \) using the distributive law if we set \( l := k \cdot C \). To sum up: For the resulting formula \( \omega_{\psi} \) we have \( \psi \not\subseteq \phi/\beta \) iff \( \beta \models \omega_{\psi} \).

It is now easy to express the first condition:

\[
\omega = \bigcup_{\psi \subseteq \rho, |\psi| \leq C, \sigma(\psi) < \delta} \omega_{\psi}
\]

and note that \( \omega \in \text{lCNFS} \) holds. Note furthermore that an AC^0 circuit can construct this \( \omega \) as \( C \) is a constant and, hence, we can consider all size-\( C \) subsets of \( \rho \) in parallel and can hardwire the tests \( \sigma(\psi) < \delta \).

For the correctness of the construction, let us briefly reiterate: First, if \( \beta \models \omega \), consider any subset \( \psi \subseteq \phi/\beta \) of size at most \( C \). Then \( \sigma(\psi) < \delta \) is impossible as \( \omega_{\psi} \) would now require that some clause \( c \in \psi \) is missing from \( \phi/\beta \). Thus, \( \sigma(\psi) \geq \delta \) always holds and, by the Threshold Locality Lemma, we have \( \sigma(\phi/\beta) \geq \delta \). Second, suppose \( \sigma(\phi/\beta) \geq \delta \). Then \( \sigma(\psi) \geq \delta \) trivially holds for every subset \( \psi \subseteq \phi/\beta \). Thus, for every \( \omega_{\psi} \) considered in \( \omega \), we have \( \psi \not\subseteq \phi/\beta \). But then, \( \beta \models \omega_{\psi} \). Thus, \( \beta \models \omega \). □
\begin{itemize}
  \item \textbf{Corollary 3.10.} MAJ-MAJ-kSAT ∈ AC^0 for all \(k\).
  \begin{proof}
  Consider the number \(l\) from the Threshold Encoding Lemma for \(k\) and \(\delta = 1/2\). We can reduce MAJ-MAJ-kSAT to ISAT-PROB_{≥1/2} by simply mapping an input formula \(\phi\) to the formula \(\varphi\) from the lemma. Since ISAT-PROB_{≥1/2} ∈ AC^0 (this is implicit in the main algorithmic results of the present section, but see Theorem 4.1 for a detailed discussion), we get the claim.
  \end{proof}

  Writing MAJ\(^i\)-kSAT for MAJ-MAJ-\cdots-MAJ-kSAT with of course exactly \(i\) repetitions (so MAJ\(^1\)-kSAT = KSAT-PROB_{≥1/2} and MAJ\(^2\)-kSAT = MAJ-MAJ-kSAT) and with the obvious semantics for \(i > 2\), we get:

  \item \textbf{Corollary 3.11.} MAJ\(^i\)-kSAT ∈ AC^0 for all \(k\) and \(i\).
  \begin{proof}
  Just repeat the reduction from the previous corollary \(i - 1\) times.
  \end{proof}

As a final remark, note that the Threshold Encoding Lemma actually provides us with yet another direct method for showing KSAT-PROB_{≥2k} ∈ AC^0: Just set \(X = \emptyset\). This means that in the proof of the above corollary, we can actually apply the reduction \(i\) times, not \(i - 1\) times, and get a “perfectly uniform” argument.

\subsection{3.3 The Recursive Interval-Bounding Algorithm}

While the sunflower-collapsing algorithm for deciding \(\sigma(\phi) \geq \delta\) gives insight into the structure of the problem and lies at the heart of the complexity-theoretic results in the next section, the starting point for a presumably more practical algorithm can be derived from the expansion operation (which we already used in the proof the Spectral Well-Ordering Theorem). The idea behind Algorithm 3 is simple: Starting with a formula \(\phi\), we wish to compute an interval \(I \subseteq [0, 1]\) with \(\sigma(\phi) \in I\) and \(|I| := \max I - \min I < \epsilon\). If we achieve this and if \(\epsilon\) is at most the spectral gap of \(\delta\), then \(\max I < \delta\) will hold iff \(\sigma(\phi) < \delta\). Depending on the structure of \(\phi\), we may be able to easily obtain an interval with the desired properties (using the method INTERVAL\((\phi)\)). If, however, the initially obtained interval is too large, by the Packing Probability Lemma it will contain a (relatively) small packing, allowing us to expand the formula and to \textit{recurse} to \((k - 1)\)CNF formulas. The following lemma makes these claims precise.

\begin{lemma}
  For each fixed \(k\) and \(\delta\), in Algorithm 3 from page 8
  \begin{enumerate}
    \item for every \(\phi\) the algorithm INTERVAL\((\phi)\) returns an interval \(I\) with \(\sigma(\phi) \in I\) and with \(|I| = 0\) when \(\phi \in 1\CNFS\),
    \item for every \(\phi\) and \(\epsilon > 0\) the algorithm BOUNDED-INTERVAL\((\phi, \epsilon)\) will
      \begin{enumerate}
        \item call itself recursively only for formulas with a strictly smaller clause size,
        \item call itself at most \(2^k \log_2 1 - 2^{-k} \epsilon\) times,
        \item return an interval \(I\) with \(\sigma(\phi) \in I\) and \(|I| < \epsilon\), and
      \end{enumerate}
    \item the output of INTERVAL-BOUNDING\((\phi)\) will be correct.
  \end{enumerate}
\end{lemma}

\begin{proof}
  For item 1, we clearly have \(\sigma(\phi) \in I\) as the interval we output is either exactly \([\sigma(\phi), \sigma(\phi)]\) (namely, when \(\phi\) is a packing and \(\sigma(\phi)\) is exactly equal to \(\prod_{\phi \in \phi} (1 - 2^{-k})\)) or is \([0, 1 - 2^{-k} |\text{PACK}(\phi)|]\) and the Packing Probability Lemma tells us that \(\sigma(\phi)\) lies in this interval. When \(\phi \in 1\CNFS\), then \(\phi\) is always a packing (unless it is contradictory as it contains both \(\{v\}\) and \(\neg v\), but this can be filtered easily) and, thus, \(|I| = 0\).
\end{proof}
For item 2a, observe that \( \text{PACK}(\phi) \) has the property that \( \text{vars(\text{PACK}(\phi))} \) intersects the variables in each clause of \( \phi \) (otherwise, \( \text{PACK}(\phi) \) would not be maximal). Thus, in all calls \( \phi|_{\beta} \) has a strictly smaller maximal clause size (unless \( \phi \in 1\text{cnfs} \) did already hold, but then there will be no recursive calls at all).

Item 2b follows as a recursive call is only made when \(|I| \geq \epsilon\), which implies that we have \( I = [0, (1 - 2^{-k})^{\text{PACK}(\phi)}] \) and thus \((1 - 2^{-k})^{\text{PACK}(\phi)} \geq \epsilon\). This clearly shows \(|\text{PACK}(\phi)| \leq \log_{1-2^{-k}} \epsilon\) and thus \(|\text{vars(\text{PACK}(\phi))}| \leq k \log_{1-2^{-k}} \epsilon\) which in turn shows that the number of different \( \beta \) used for the recursive calls is the claimed number.

Item 2c follows by induction on \( k \): By the first item, if \( \phi \in 1\text{cnfs} \), then \(|I| = 0 < \epsilon\). For the inductive step, for any \( \phi \in k\text{cnfs} \) we have \( \sigma(\phi) = \sum_{\beta} \sigma(\phi|_{\beta}) \). By the induction hypothesis, our algorithm will compute for each \( \phi|_{\beta} \) an interval \( I_{\beta} \) containing \( \sigma(\phi|_{\beta}) \). Then the sum of these intervals will contain \( \sigma(\phi) \). Furthermore, the size of the sum of these intervals will be the sum of the sizes of the intervals. As the size of each \( I_{\beta} \) at most \( \epsilon \) divided by the number of calls made, the total interval size at most \( \epsilon \).

The last item now follows as an interval \( I \) with \(|I| < \text{spectral-gap}_{k\text{cnfs}}(\delta)\) and \( \sigma(\phi) \in I \) has the following property: If \( \sigma(\phi) < \delta \), then \( \sigma(\phi) \leq \delta - \text{spectral-gap}_{k\text{cnfs}}(\delta) \) and thus \( \sigma(\phi) < \delta - |I| \) or, equivalently, \( \sigma(\phi) + |I| < \delta \). In particular, max \( I < \delta \), which is exactly what we test. In contrast, if \( \sigma(\phi) \geq \delta \) we immediately get max \( I \geq \delta \). Thus, the output is always correct.

4 Complexity-Theoretic Results

We prove Theorem 1.11, which tells us precisely for each \( \delta \in [0, 1] \) whether \( \text{ksat-prob}_{\geq \delta} \) is \( \text{NP} \)-complete, \( \text{NL} \)-complete, or in \( \text{AC}^0 \), in three steps: First, we show that \( \text{ksat-prob}_{\geq \delta} \) lies in \( \text{AC}^0 \) for all \( k \) and \( \delta \). This means that from a complexity-theoretic point of view it will always be “trivial” to check whether \( \sigma(\phi) \geq \delta \) holds – the tricky part is showing that \( \sigma(\phi) \neq \delta \) then holds. Second, we introduce a technical definition of being a \( k \)-target for \( k\text{cnfs} \), which is set up in such a way that if a number \( \delta \) has this property, it will be possible to reduce \( \text{lsat} \) to \( \text{ksat-prob}_{\geq \delta} \). This will prove the \( \text{NL} \)- and \( \text{NP} \)-hardness results for those \( \delta \) where Theorem 1.11 claims that \( \text{ksat-prob}_{\geq \delta} \) is \( \text{NL} \)- or \( \text{NP} \)-complete. Third, we address the upper bounds, meaning that we present \( \text{NP} \)-, \( \text{NL} \)-, and \( \text{AC}^0 \)-algorithms that match the lower bounds. Here, we show that the variables in the kernel form backdoor sets for the “error the kernelization makes” into \( 2\text{cnfs} \) or \( 1\text{cnfs} \), when \( \delta \) is not a \( k \)-target for \( 3\text{cnfs} \) or \( 2\text{cnfs} \), respectively. Figure 4 is an attempt to visualize the underlying ideas in a simplified way.

4.1 Upper Complexity Bounds for the Greater-or-Equal Problem

Although our ultimate objective in this section is the complexity of the problem of telling whether \( \sigma(\phi) > \delta \) holds, we begin with the following:

\begin{itemize}
  \item \textbf{Theorem 4.1.} For all \( k \) and \( \delta \) the problem \( \text{ksat-prob}_{\geq \delta} \) lies in \( \text{dlogtime-uniform AC}^0 \).
\end{itemize}

\textbf{Proof.} We claim that Algorithm 1 can be implemented by \( \text{AC}^0 \) circuits. It is well-established that kernels for hitting sets can be computed by (\text{dlogtime-uniform}) \( \text{AC}^0 \)-circuits parameterized by the size of the hitting set and the size of the hyperedges, see \cite{3, 4}. Translated to our setting, these algorithms compute the core \( \kappa \) from Algorithm 1 for positive \( \phi \) in \( \text{AC}^0 \) when parameterized by the maximum clause size \( k \) and the size \( h \) of the to-be-collapsed sunflowers. As the algorithm can easily be adapted to cope with the fact that sunflowers for formulas must take the “signs” of the literals in the cores into account, see the discussion prior to Corollary 3.2, we get the claim as both \( k \) and \( h \) are fixed.

\end{itemize}
The input formula $\phi \in 5\text{cnfs}$

The kernel $\kappa \in 5\text{cnfs}$

The input formula $\phi \in 5\text{cnfs}$

The kernel $\kappa \in 5\text{cnfs}$

Using $\text{vars}(\kappa)$ as a backdoor into 2CNFS

Reducing 2SAT to 5SAT-PROB$_{>_\sigma}(\kappa)$

Figure 4: Continuing the example from Figure 2 from page 5: At the top, a formula $\phi \in 5\text{cnfs}$ is shown together with a kernel $\kappa$ obtained by collapsing the red link to the dashed green clause $c = \{x, \neg y, z\}$. The size 3 of $c$ is important: First, it is the minimum size of clauses in the kernel, which implies that if we remove all variables in the kernel from the clauses of $\phi$, we get the lower left figure in which the blue clauses all have size at most $2 = 5 - 3$. In particular, $\phi|_{\beta}$ for $\beta: \text{vars}(\kappa) \to \{0, 1\}$ lies in 2CNFS and $\text{vars}(\kappa)$ is a backdoor into 2CNFS. Second, the clause $c$ with $|c| \leq 3$ has "room left for two literals," meaning that for $\psi \in 2\text{CNFS}$ we have $\{c \cup d \mid d \in \psi\} \in 5\text{CNFS}$. The lower right figure shows what happens when we add the clauses of $\psi$ (assuming $\text{vars}(\psi) = \{p_1, \ldots, p_n\}$, these new variables are colored red) to the kernel by "adding them to $c$" (the dashed clauses). The resulting orange formula will have a satisfaction probability strictly larger than that of $\kappa$ iff $\psi$ is satisfiable. Note, however, that instead of $\kappa$ we may need to use a related formula $\omega$ instead, see equation (3) on page 24 for details, namely when the clauses in $\kappa$ other than $c$ already imply $c$ (which they do not in the example).
This theorem can be thought of as a strengthened version of the result by Akmal and Williams [2] that $k\text{SAT-PROB}_{>\delta} \in \text{LINTIME}$ holds for all $k$ and $\delta$, but two remarks are in order: First, the algorithm of Akmal and Williams can actually also be turned into an $\text{AC}^0$ algorithm [17], meaning that the result is already implicit in their work. Second, the existence of an $\text{AC}^0$-circuit family for a problem does not necessarily imply that the problem can be solved in linear time—one needs an $\text{AC}^0$ circuit family of linear size for this. It is, however, not clear whether such a family exists as the natural way of searching for packings or sunflowers involves color coding, which seems to need a quadratic size for derandomization. While these considerations will not be important for the complexity-theoretic statements of the main theorem, further research on the complexity of $k\text{SAT-PROB}_{>\delta}$ could address these questions.

4.2 Lower Complexity Bounds for the Strictly-Greater-Than Problem

We now show that $k\text{SAT-PROB}_{>\delta}$ is NL- or NP-hard for certain values of $k$ and $\delta$. For this, we need the already mentioned notion of “$k$-targets for $l\text{CNFS}$” whose definition was as follows (recall $\omega_s = \{c \in \omega \mid |c| > s\}$ and $\omega_{\leq s} = \{c \in \omega \mid |c| \leq s\}$):

Definition 4.2. For numbers $k$ and $l$, we say that a number $\delta \in [0,1]$ is a $k$-target for $l\text{CNFS}$ if $\delta = \sigma(\omega)$ for some $\omega \in l\text{CNFS}$ with $\omega_{>k-1} \neq \omega_{\leq k-1}$.

An example was $\delta = 7/32$, which is a 3-target for 2CNFS as demonstrated by $\omega = \\{\{a\}, \{b\}, \{c_1, c_2, c_3\}\}$ since $\omega_{>1-2} = \\{\{c_1, c_2, c_3\}\}$ and $\omega_{\leq 3-2} = \omega_{\leq 3}$. The term “target” is motivated by the following lemma: The formulas $\omega$ can serve as “targets for a reduction from $l\text{SAT}$”, which proves the lower complexity bounds for $k\text{SAT-PROB}_{>\delta}$ from Theorem 1.11.

Lemma 4.3. If $\delta$ is a $k$-target for $3\text{CNFS}$, then $k\text{SAT-PROB}_{>\delta}$ is NP-hard.

Proof. We reduce 3SAT to $k\text{SAT-PROB}_{>\delta}$: Let $\omega$ witness that $\delta$ is a $k$-target for 3CNFS and let $\beta$: $\text{vars}(\omega) \rightarrow \{0,1\}$ witness $\omega_{>k-3} \neq \omega_{\leq k-3}$ (that is, $\beta$ is a model of $\omega_{>k-3}$ but not of $\omega_{\leq k-3}$). Let $\psi \in 3\text{CNFS}$ be an input for the reduction, that is, we wish to reduce the question of whether $\sigma(\psi) > 0$ holds to the question of whether $\sigma(\rho) > \delta$ holds for some $\rho \in k\text{CNFS}$. If necessary, rename the variables in $\psi$ to ensure $\text{vars}(\psi) \cap \text{vars}(\omega) = \emptyset$, and map $\psi$ to $\rho = \omega_{>k-3} \cup \{c \cup d \mid c \in \omega_{<k-3}, d \in \psi\}$.

We claim $\sigma(\rho) > \delta$ if $\psi \in 3\text{SAT}$. For the first direction, assume $\sigma(\rho) > \delta = \sigma(\omega)$. As $\omega \models \rho$ (every clause of $\rho$ is a superclause of some clause of $\omega$), there must exist an assignment $\gamma$ with $\gamma \models \rho \supset \omega_{>k-3}$ and $\gamma \not\models \omega_{\leq k-3}$. This means that there must be a clause $\beta \in \omega_{\leq k-3}$ with $\gamma \not\models \beta$. However, $\gamma \models \rho \supset \{c^* \cup d \mid d \in \psi\}$, which implies $\gamma \models \psi$ and $\psi \in 3\text{SAT}$.

Second, assume $\psi \in 3\text{SAT}$ and let $\alpha$: $\text{vars}(\psi) \rightarrow \{0,1\}$ witness $\psi \in 3\text{SAT}$. Consider the assignment $\gamma$: $\text{vars}(\psi) \cup \text{vars}(\omega) \rightarrow \{0,1\}$ defined by $\gamma(\beta) = \alpha(\beta)$ for $\beta \in \text{vars}(\psi)$ and $\gamma(\gamma) = \beta(\gamma)$ for $\gamma \in \text{vars}(\omega)$. Trivially, $\gamma \not\models \beta$ as $\beta \not\models \omega_{\leq k-3}$. Each clause $c \in \omega_{>k-3}$ is satisfied as $\beta \models \omega_{>k-3}$. Each clause $c \cup d \in \rho$ with $d \in \psi$ is satisfied as $\alpha \models \psi$ and $d \in \psi$. In total, $\gamma$ is a model of all clauses of $\rho$ and thus $\sigma(\rho) > \sigma(\omega) = \delta$.

Lemma 4.4. If $\delta$ is a $k$-target for 2CNFS, then $k\text{SAT-PROB}_{>\delta}$ is NL-hard.

Proof. The proof is identical to that of the previous lemma, only we reduce from 2SAT.

In Figure 4 in the lower right subfigure the effect of adding the clauses of a formula $\psi \in 2\text{CNFS}$ to the small clauses of an example formula $\kappa = \omega$ is shown (although there is only a single small clause $c$ in the example).
4.3 Upper Complexity Bounds for the Strictly-Greater-Than Problem

To complete the proof of Theorem 1.11, we need to prove the upper bounds. To get some intuition, let us start with the easiest case, the upper bound of NP, that is, the claim that \( k\text{-SAT-PROB}_{>\delta} \in \text{NP} \) always holds. Recall that in Theorem 4.1 we showed \( k\text{-SAT-PROB}_{\geq\delta} \in \text{AC}^0 \) by kernelizing input formulas \( \phi \): We replaced the question of whether \( \sigma(\phi) \geq \delta \) holds by the question of whether \( \sigma(\kappa) \geq \delta \) holds – and \( \sigma(\kappa) \) can be computed by brute force. However, this is not directly helpful for deciding whether \( \sigma(\phi) > \delta \) holds: If \( \sigma(\kappa) > \delta \), we also know that \( \sigma(\phi) > \delta \) holds; if \( \sigma(\kappa) < \delta \), we also know that \( \sigma(\phi) < \delta \) holds (because of the spectral gap); but if \( \sigma(\kappa) = \delta \), both \( \sigma(\phi) = \delta \) and \( \sigma(\phi) > \delta \) are still possible. The “critical” case \( \sigma(\kappa) = \delta \) forces us to investigate further: We must find out whether the sunflower collapsing process “destroyed solutions,” that is, whether there is an assignment \( \beta : \text{var}(\phi) \rightarrow \{0, 1\} \) with \( \beta \models \phi \), but \( \beta \not\models \kappa \). Fortunately, this is easy to do using an NP-machine: We can just guess such an assignment. Let us state this observation as a lemma for future reference:

Lemma 4.5. For all \( k \) and \( \delta \), we have \( k\text{-SAT-PROB}_{>\delta} \in \text{NP} \).

For the NL upper bound, we can basically proceed the same way – we just need a way to determine whether there exists a \( \beta \) with \( \beta \models \phi \), but \( \beta \not\models \kappa \) using an NL-machine. An idea towards achieving this is to simply iterate over all possible \( \beta \): \( \text{var}(\kappa) \rightarrow \{0, 1\} \) with \( \beta \not\models \kappa \) and then test whether \( \phi|_{\beta} \) is satisfiable. Of course, we still have to answer (many) questions of the form “\( \phi|_{\beta} \in \text{SAT} \)?” – but we may now hope that \( \phi|_{\beta} \in 2\text{CNFS} \) might hold: After all, clauses in \( \phi|_{\beta} \) result from removing all variables in the core from the clauses of \( \phi \) and a large core thus means small clauses in \( \phi|_{\beta} \), see the lower left part of Figure 4 for a concrete example. If we always had \( \phi|_{\beta} \in 2\text{CNFS} \), then \( \text{vars}(\kappa) \) would be called a backdoor set into \( 2\text{CNFS} \) and this would suffice to show \( k\text{-SAT-PROB}_{>\delta} \in \text{NL} \).

Now, the variables in the kernel do not always form a backdoor set into \( 2\text{CNFS} \), but we will be able to construct a formula \( \omega \) for which they do and this formula will serve as a “replacement” for \( \phi \): If it is not equivalent to \( \phi \), then either \( \sigma(\phi) > \sigma(\omega) > \delta \) will follow immediately or \( \delta \) is a \( k \)-target for \( 3\text{CNFS} \). As the latter is ruled out by the assumption of the theorem, we will have \( \sigma(\phi) > \delta \) if \( \sigma(\omega) > \delta \). Thus, having a backdoor set in \( 2\text{CNFS} \) for \( \omega \) will suffice to show \( k\text{-SAT-PROB}_{>\delta} \in \text{NL} \). The details follow.

Backdoor Sets. Backdoor sets are a powerful tool from FPT theory [18] with a rich theory around them; but for our purposes only the following kind will be important:

Definition 4.6. Let \( \Delta \subseteq \text{CNFS} \) and let \( \psi \in \text{CNFS} \). A (strong) backdoor set for \( \psi \) into \( \Delta \) is a set \( V \) of variables such that for all \( \beta : V \rightarrow \{0, 1\} \) we have \( \psi|_{\beta} \in \Delta \).

The importance of backdoor sets in FPT theory lies, of course, in the fact that in order to determine whether \( \psi \) is satisfiable, it suffices to find a \( \beta : V \rightarrow \{0, 1\} \) such that \( \psi|_{\beta} \) is satisfiable. If \( \Delta \) is a tractable set (like \( \Delta = 2\text{CNFS} \)), this allows us to efficiently decide \( \psi \in \text{SAT} \) as long as the backdoor set is not too large. In our context, the variables in the kernel will form a backdoor set into \( 2\text{CNFS} \) or \( 1\text{CNFS} \), not for the original formula \( \phi \), but only for an “intermediate” formula \( \omega \), defined as follows.

The Replacement Formula. Fix \( k \) and \( l \) and fix a formula \( \phi \in k\text{CNFS} \). Let \( \kappa \) denote the kernel computed in Algorithm 1 on input \( \phi \) (that is, the result of collapsing the links of cores of sunflowers large enough to ensure that the collapse step reduces the satisfaction probability by less than the spectral gap of \( \delta \), which ensures by the No Tunneling Lemma that \( \sigma(\phi) \) “stays above \( \delta \), if it was above \( \delta \)” and assume that we are in the critical case \( \delta = \sigma(\kappa) \) where
“there is still work to be done.” Define \( \omega \in k\text{-CNFS} \) as follows (with \( \kappa_{\leq s} = \{ c \in \kappa \mid |c| < s \} \) and \( \kappa_{\geq s} = \{ c \in \kappa \mid |c| \geq s \} \): and note that \( \kappa_{\leq s} = \kappa_{\leq s-1} \) and \( \kappa_{\geq s} = \kappa_{\geq s-1} \), so the subscripts of \( \kappa \) are “shifted by \(-1\)” relative to the subscripts of \( \omega \) in Definition 4.2):

\[
\omega := \kappa_{<k-1} \cup \bigcup_{c \in \kappa_{\geq k-1}} \text{link}_\phi(c). \tag{3}
\]

The importance of \( \omega \) lies in the following two lemmas:

> **Lemma 4.7 (Backdoor Lemma).** The set \( \text{vars}(\kappa) \) is a backdoor set for \( \omega \) into \( k\text{-CNFS} \).

**Proof.** For any \( \beta : \text{vars}(\kappa) \to \{0,1\} \), consider any clause \( e \in \omega|_\beta \). We wish to show \( |e| \leq l \). By definition of \( \omega|_\beta \), \( e \) resulted from taking some clause in \( d \in \omega \) and stripping away all occurrences of variables in \( \text{vars}(\kappa) \). If \( d \in \kappa_{<k-1} \subseteq \kappa \), we would have \( e = 0 \) and \( |e| = 0 \). If \( d \in \text{link}_\phi(c) \) for some \( c \in \kappa_{\geq k-1} \), then \( |e| = |d \setminus c| = |d| - |c| \leq k - (k - l) = l \).

The Backdoor Lemma tells us that we can “handle” \( \omega \) well for \( l = 2 \) and \( l = 1 \). The question is, of course, how, exactly, \( \omega \) is related to \( \phi \) and \( \kappa \). The next lemma tells the story:

> **Lemma 4.8.** One of the following always holds:

1. \( \delta = \sigma(\kappa) = \sigma(\omega) = \sigma(\phi) \) or
2. \( \delta = \sigma(\kappa) < \sigma(\omega) = \sigma(\phi) \) or
3. \( \delta = \sigma(\kappa) < \sigma(\omega) < \sigma(\phi) \) or
4. \( \delta \) is a \( k\)-target for \((l+1)\text{-CNFS}\).

**Proof.** As we clearly have \( \kappa \models \omega \models \phi \) (that is, every model of \( \kappa \) is also a model of \( \omega \), and \( \omega \)'s models are models of \( \phi \)), we have \( \delta = \sigma(\kappa) \leq \sigma(\omega) \leq \sigma(\phi) \). This means that the only possibility not covered by the first three items is \( \delta = \sigma(\kappa) = \sigma(\omega) < \sigma(\phi) \) and we must show that this implies that \( \delta \) is a \( k\)-target for \((l+1)\text{-CNFS}\). For this we must show that there is an assignment \( \beta \) with \( \beta \not\models \omega_{\leq k-l-1} \) and \( \beta \models \omega_{> k-l-1} \).

If \( \sigma(\omega) < \sigma(\phi) \), there must be a model \( \beta \models \phi \) with \( \beta \not\models \omega \) and we claim that this model is a witness for \( \delta \) being a \( k\)-target. To see this, first note that \( \beta \models \phi \) and \( \beta \not\models \omega \) implies \( \beta \not\models \omega \setminus \phi \) and \( \omega \setminus \phi \subseteq \omega_{<k-1} \) by (3). Thus, \( \beta \not\models \omega_{<k-1} = \omega_{<k-1} \). Second, \( \beta \models \omega_{> k-l-1} = \omega_{> k-l-1} \) is implied by \( \beta \models \phi \) together with \( \phi \supseteq \omega_{> k-l-1} \) as every clause of size \( k - l \) or larger in \( \omega \) comes from a link and links are subsets of \( \phi \).

The Algorithm and Its Correctness. We are now ready to assemble the ideas into an algorithm, Algorithm 4, and prove its correctness.

> **Lemma 4.9.** Let \( \delta \) not be a \( k\)-target for \((l+1)\text{-CNFS}\). Then for each \( \phi \in k\text{-CNFS} \) the output of \text{REDUCE-TO-SAT}(\phi) from Algorithm 4 will be correct.

**Proof.** The output of the then-clause of the first if-statement in line 3 is correct as we trivially have \( \sigma(\kappa) \leq \sigma(\phi) \) (replacing a sunflower by its core can only reduce the satisfaction probability). The output of the then-clause of the second if in line 4 is also correct as \( \sigma(\kappa) \geq \delta \) iff \( \sigma(\phi) \geq \delta \) holds by Lemma 3.3.

By Lemma 4.8, for \( \omega \) computed in line 6 one of the following holds (the fourth possibility is ruled out by the assumption):

1. \( \delta = \sigma(\kappa) = \sigma(\omega) = \sigma(\phi) \) or
2. \( \delta = \sigma(\kappa) < \sigma(\omega) = \sigma(\phi) \) or
3. \( \delta = \sigma(\kappa) < \sigma(\omega) < \sigma(\phi) \).
Algorithm 4 An algorithm for deciding $k\text{SAT-PROB}_{\varphi,\delta}$ that is correct if $\delta$ is not a $k$-target for $(l + 1)$CNFS: In this case, Lemma 4.8 states that $\omega$ is a suitable replacement for $\phi$ and Lemma 4.7 states that the variables in the kernel form a backdoor set into $l$CNFS for $\omega$, allowing us to perform the tests in line 8 by an NL machine for $l = 2$, and even by $AC^0$-circuits for $l = 1$.

```plaintext
algorithm REDUCE-TO-SAT(\phi) // \phi \in k\text{CNFS} must hold, l is a number
1: \kappa \leftarrow \text{KERNELIZE}(\phi, \log_{l+2-\delta}(\text{spectral-gap}_{l\text{CNFS}}(\delta)))
2: \quad \text{if } \sigma(\kappa) > \delta \text{ then return } \text{"}\sigma(\phi) > \delta\text{"} 
3: \quad \text{if } \sigma(\kappa) < \delta \text{ then return } \text{"}\sigma(\phi) < \delta\text{"} 
4: \quad \quad \text{// Critical case: } \sigma(\kappa) = \delta 
5: \quad \quad \omega \leftarrow \kappa_{<k-l} \cup \bigcup_{c \in \kappa_{>k-l}} \text{link}_c(\kappa)
6: \quad \quad \text{foreach } \beta : \text{vars}(\kappa) \to \{0, 1\} \text{ with } \beta \not\models \kappa \text{ do}
7: \quad \quad \quad \text{if } \omega|_\beta \in \text{iSAT} \text{ then}
8: \quad \quad \quad \quad \text{// we now know } \sigma(\omega) > \sigma(\kappa) = \delta
9: \quad \quad \quad \quad \text{\quad return } \text{"}\sigma(\phi) > \delta\text{"} 
10: \quad \quad \quad \text{// we now know } \sigma(\omega) = \sigma(\kappa) = \delta 
11: \quad \quad \quad \text{\quad return } \text{"}\sigma(\phi) = \delta\text{"} 
12: \quad \quad \text{return } \text{"}\sigma(\phi) = \delta\text{"}
```

In particular, if $\delta = \sigma(\kappa) = \sigma(\omega)$, we know that $\delta = \sigma(\phi)$ must also hold; and if $\sigma(\kappa) < \sigma(\omega)$ we trivially have $\delta < \sigma(\phi)$. In other words:

$$\sigma(\omega) > \delta \iff \sigma(\phi) > \delta. \quad (4)$$

In the main for-loop of the algorithm, we check whether there is a model of $\omega$ that is not a model of $\kappa$. Clearly, this is the case iff there is some $\beta : \text{vars}(\kappa) \to \{0, 1\}$ with $\beta \not\models \kappa$ and $\omega|_\beta \in \text{iSAT}$. However, since by Lemma 4.7 we know that $\text{vars}(\kappa)$ is a backdoor for $\omega$ into $l$CNFS, it is correct to test only $\omega|_\beta \in \text{iSAT}$ inside the loop.

All told, when the comment lines 9 or 11 are reached, the comments’ statements are correct. By (4) this means that the two outputs in the subsequent lines are correct. ▶

The Upper Bounds. We can now prove the three upper bounds from Theorem 1.11 and thus prove the Spectral Trichotomy Theorem, Theorem 1.6 from the introduction. The NP upper bound has already been stated in Lemma 4.5. The argument for NL is as follows:

▶ **Lemma 4.10.** If $\delta$ is not a $k$-target for $3$CNFS, then $k\text{SAT-PROB}_{\varphi,\delta} \in \text{NL}$.

**Proof.** By Lemma 4.9, Algorithm 4 will correctly decide $k\text{SAT-PROB}_{\varphi,\delta}$ in this case. To see that the algorithm can be implemented by an NL-machine, observe that the for-loop iterates only over a constant number of $\beta$ (the kernel size is fixed) and can thus be hardwired into the machine. The central test $\omega|_\beta \in 2\text{SAT}$ clearly only requires an NL machine. ▶

▶ **Lemma 4.11.** If $\delta$ is not a $k$-target for $2$CNFS, then $k\text{SAT-PROB}_{\varphi,\delta} \in \text{AC}^0$.

**Proof.** Just as in the previous corollary, we invoke Lemma 4.9 and the for-loop still iterates only over a constant number of $\beta$. The central test is now $\phi|_\beta \in \text{1SAT}$, which is possible to perform using $\text{AC}^0$ circuits. ▶

5 Conclusion

The results of the present paper settle the complexity of $k\text{SAT-PROB}_{\varphi,\delta}$ from a complexity-theoretic point of view: The problem is either NP-complete or NL-complete or lies in $\text{AC}^0$ – and which of these is the case depends on whether or not $\delta = \sigma(\omega)$ holds for some
On the Satisfaction Probability of \(k\)-CNF Formulas

A formula \(\omega\) with certain syntactic properties. The proof is based on the insight that the spectra \(k\)CNFS-\(\sigma\)-SPECTRUM are well-ordered with respect to \(>\), as this implies (1) that the standard sunflower-based kernel algorithm for hitting sets allows us to compute kernels for \(k\text{SAT-PROB}_{\geq \delta}\) and (2) that the variables in the kernel form strong backdoor sets into 2CNFS or 1CNFS for formulas whose satisfaction probabilities “behave the same way” as those of the input formula.

An attempt to visualize the “landscape” of the complexity of \(k\text{SAT-PROB}_{>\delta}\) for \(k \leq 4\) can be found in Figure 1 on page 2. For \(k = 4\), two values of special interest are \(\delta_1 = 15/32 = \frac{1}{2} - \frac{1}{16}\) and \(\delta_2 = 63/128 = \frac{1}{2} - \frac{1}{128}\). There is a “red triangle” in the figure at \(\delta_1\), meaning that \(4\text{SAT-PROB}_{>15/32}\) is \(\text{NP}\)-complete by Theorem 1.11 as \(15/32 = \sigma(\{\{a\}, \{x, y, z, w\}\})\). There is a “green triangle” at \(\delta_2\) (as well as at many, many other positions in \((15/32, 1/2)\), but still only at a nowhere dense subset despite the “solid line” in the visualization) as \(4\text{SAT-PROB}_{>63/128}\) is \(\text{NL}\)-complete since \(63/128 = \sigma(\{\{a, b\}, \{c, d\}, \{e, f, g\}\})\), but no 4CNF formula \(\phi\) containing a singleton clause has \(\sigma(\phi) = 63/128\).

For larger values of \(k\), observe that, on the one hand, \(k\text{SAT-PROB}_{>15/32}\) is \(\text{NL}\)-complete for all \(k\) (since \(\sigma(\{\{a_1, \ldots, a_{k-2}\}\}) = 1 - 2^{-(k-2)}\), but \(\sigma(\omega) \neq 1 - 2^{-(k-2)}\) for all \(\omega \in k\text{CNFS}\) containing a clause of size \(k - 3\) as this clause already lowers the satisfaction probability to at most \(1 - 2^{-(k-3)} < 1 - 2^{-(k-2)}\); while on the other hand, \(k\text{SAT-PROB}_{>63/128}\) is \(\text{NP}\)-complete for all \(k \geq 4\) and \(i \geq i\).

In addition to the “strictly greater than” problem \(k\text{SAT-PROB}_{>\delta}\) and the “boring” problem \(k\text{SAT-PROB}_{\geq \delta}\) (which is always in \(\text{AC}^0\)), one can also consider the “equal to” version. Combining the results from this paper immediately yields: For the same \(k\) and \(\delta\) as in Theorem 1.11, the problem \(k\text{SAT-PROB}_{=\delta}\) is \(\text{coNP}\)-complete, \(\text{NL}\)-complete, or lies in \(\text{AC}^0\).

Spelled out, we get results like the following: “it is \(\text{NL}\)-complete to decide on input of a 3CNF formula whether exactly half of the assignments are satisfying” and “it is \(\text{coNP}\)-complete to decide on input of a 4CNF formula whether exactly half of the assignments are satisfying,” but also stranger ones like “it is \(\text{NL}\)-complete to decide on input of a 4CNF formula whether the fraction of satisfying assignments is exactly \(\frac{1}{2} - \frac{1}{128}\)” while “it is \(\text{coNP}\)-complete to decide on input of a 4CNF formula whether the fraction of satisfying assignments is exactly \(\frac{1}{2} - \frac{1}{32}\).”

Since the algorithms presented in this paper depend so heavily on the size of spectral gaps, it is of interest to determine these sizes precisely. While explicit bounds can be shown \([14]\), it is very much unclear whether these hyperexponentiation bounds are even remotely tight. It would also be of interest to determine explicit values: A close look at Figure 1 reveals \(\text{spectral-gap}_{2\text{CNFS}}(1/2) = 1/32\), but what is the value of \(\text{spectral-gap}_{3\text{CNFS}}(1/2)\)?

On the one hand, the results of the present paper “settle” the complexity of many versions of satisfaction probability threshold problems for \(k\text{CNFS}\), including the majority-of-majority version; on the other hand, many new problems arise. These include questions concerning satisfaction probability problems for constraint satisfaction problems, questions concerning the fixed-parameter tractability of satisfaction probability problems, questions surrounding the complexity of algebraic representations of formulas, a whole bunch of questions arising from logical and descriptive reformulations, and finally – of course – questions concerning practical implementation. In the technical report version \([14]\) some first ideas and partial answers to these questions are presented, but there is certainly still much to do.

References


